# I/O NEWS

*Trapping Bad Blocks*
*Fortran-88 Standard*
*SDI & SGS Graphics*

THE OFFICIAL PUBLICATION OF THE INTERNATIONAL ASSOCIATION OF CROMEMCO USERS

Volume Six, Number Three

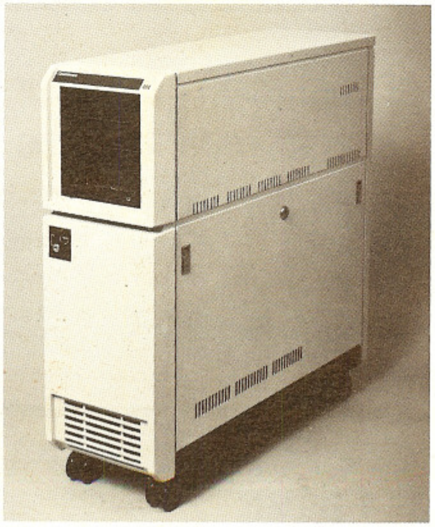
Single Copy Price $10.00

## Cromemco Introduces High-Performance CS-460

Cromemco began 1988 with the introduction of a new high-performance system, called the CS-460, that has more processing power and storage capacity than any other computer in Cromemco's line.

The central processor of the CS-460 is a special, high-performance version of the highly regarded 32-bit XXU processor card. This new processor, called the XXU-25, includes a 68881 math coprocessor chip that operates at a blazing 25MHz clock rate. The CS-460 benchmarks at over 1.3 million Whetstones per second. This is a 30% speed improvement over its progenitor, the CS-420.

The new XXU-25 also has an expanded version of on-board XDOS software. With this new XDOS, the operator can change the default port for the system console, which was not possible with earlier Cromemco systems. XDOS also now has expanded support for cartridge tape drives, including the ability to boot the system directly from cartridge tape.



The cartridge tape drive built into the CS-460 has an enormous 120 megabyte capacity. The new tape drive uses the industry standard QIC-120 tape format. The tape drive can, however, read tapes in either the QIC-120 or QIC-24 format, assuring compatibility with tapes written on Cromemco's CTD-60 tape drive. (Tapes written on Cromemco's older CTD tape drive, though, cannot be read by the CS-460.)

The CS-460 includes the UNIX System V.2 operating system installed on the system's 175 megabyte hard disk drive. A backup copy of the operating system is supplied on cartridge tape. For added convenience, Cromemco now offers all of its 68020-based UNIX software on tape cartridges. Cartridge-based software is specified by a "Q" suffix in the software's model number. (ISQL-Q, for example, is the model number for Informix Structured Query Language on cartridge tape.)

One surprise with the CS-460 is that there is *no* floppy disk drive. Cromemco believes that with the length and complexity of today's UNIX software, the cartridge tape will replace the floppy disk as the preferred method for both backup and for programs and data interchange. Certainly, anyone who has had to reload UNIX from two or three boxes full of floppy diskettes will agree that this is a trend in the right direction!

The CS-460 comes with either one or two 175 megabyte disk drives. The drives interface to Cromemco's new high-performance ESDC interface card (*I/O NEWS, Vol. 6, No. 2*). Cromemco expects to announce optional 380 megabyte and 760 megabyte disk drives for the CS-460 later this year. With dual 760 megabyte drives, the CS-460 will have an awesome 1580 megabytes of internal hard disk storage!

*Continued on page 15*

## dBIII Compiler Revisited

*by Charles M. Perrella*

During the summer of 1986 members of our firm reviewed the first release of dbIII Compiler. This review was published in the *I/O NEWS, Volume 5, Number Four*. Considering the extra power the new Version 2 has and the rise of dBIII+ as the multi-user standard of data base management for the MS-DOS world, I felt an updated review of this package was in order.

dBIII Compiler, written by WordTech and ported to Cromix and Unix by Software Standards, Inc., is a multi-user compiler that enables the running of most dBIII+ code on a Cromemco machine. Written to run in a 68000 environment, it has all of the necessary locking schemes to make it a very good multi-user data base system.

*Continued on page 15*

## Winning at Copyrights

*by Paul Hentzel*

**International Copyright Matters**

In the United States, the first and most fundamental step towards protecting your copyright interest in software (and all related Works) is to place a copyright notice directly on all copies of your software (and on every copyrightable Work that relates to your software). This U.S. copyright notice must include:

1. The circle C symbol © (or the word copyright),
2. The year of first publication, and,

*Continued on page 19*

# A few words say it all...

"Frankly, there is very little left that could be done to improve ProCall." Greg Pepper, I/O NEWS, Volume IV, Number III

"...ProCall fills a significant need in the computer communications area" Ron Blaylock, I/O NEWS, Volume V Number II

"Actually, it is the best Users Manual I have ever read..." Bill Jaenicke, I/O NEWS, Volume V Number VI

From the original ProCall for CDOS and CROMIX, to ProCall/PC for personal computers, to the new ProCall/V for UNIX and XENIX, there is no better communications software for Cromemco users.

| | |
|---|---|
| **ProCall/V** for Cromemco UNIX (avail. soon) | 495.00 |
| **ProCall/S100** for CDOS, CROMIX, CRO-D | 349.00 |
| **ProCall/PC** ***plus*** for DOS & OS/2 PCs | 249.00 |
| **ProCall/C-10** for Cromemco C-10 | 195.00 |

**C-20 Owners! Send us a copy of your Cromemco sales slip to save $100 off your copy of ProCall/PC *plus.***

PROVE IT TO YOURSELF! See your dealer now for a demonstration or call our Sales department at (408) 263-8665. We'll send you data sheets, reprints of the above reviews, and a coupon for up to $50 off your own copy of ProCall (authorized dealers also carry the coupon).

PROCALL

PSD
1145 Park Heights Drive
Milpitas, CA 95035-4605 USA

# table of contents

Jan/Feb 1988

Volume Six, Number Three

# I/O NEWS

*The Official Publication of The International Association of Cromemco Users is available through membership in the association. Editorial and advertising policies are designed for the enlightenment of the members in regard to new uses for, and developments of, Cromemco products and other products compatible with Cromemco systems.*

I/O NEWS [ISSN 0274-9998] is published bi-monthly by The International Association of Cromemco Users (a California corporation), 24843 Del Prado, Suite 473, Dana Point, CA 92629-2852. General Offices are at 34021 Granada #B, Dana Point, CA 92629. Telephone: (714) 661-9764. Second Class Postage Paid at Dana Point and Santa Ana, CA.
**POSTMASTER: Send address changes to I/O NEWS, 24843 Del Prado, Suite 473, Dana Point, CA 92629-2852.**
Subscriptions to I/O NEWS are entered with membership in The IACU. Yearly memberships may be purchased for $42 (U.S. delivery address), $49 (delivery address in Canada or Mexico), and $60 (other international delivery addresses). Contact IACU for multi-year membership rates. Back issues of I/O NEWS are available for $10.00 per issue. Please note: all prices are in U.S. dollars. Return postage must be included with all manuscripts and photos submitted if they are to be returned. The IACU and I/O NEWS accept no responsibility for the return of unsolicited materials.
All rights in letters sent to IACU and I/O NEWS will be treated as unconditionally assigned for publication and copyright to comment editorially and to edit.

William E. Jaenicke
Editor and Publisher
Lisa B. Jaenicke
Business Manager

Art Direction
Howard Millman

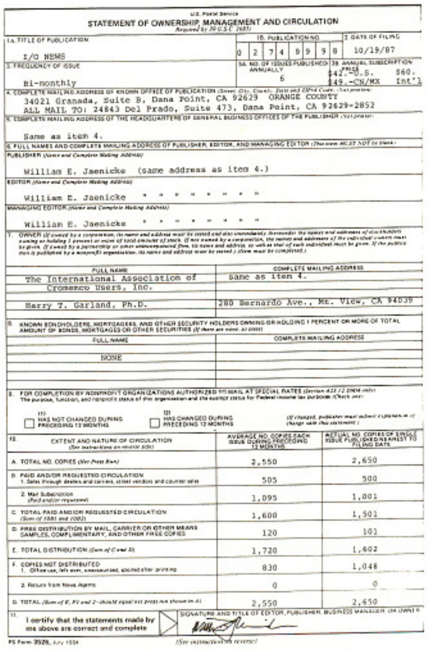
Printing
Hallmark Litho

Typography
Western Outdoors

## COVER FEATURES

## ARTICLES & FEATURES

## DEPARTMENTS



U.S. Postal Service
STATEMENT OF OWNERSHIP, MANAGEMENT AND CIRCULATION
*Required by 39 U.S.C. 3685*

1A. TITLE OF PUBLICATION
I/O NEWS

1B. PUBLICATION NO.
0 2 7 4 9 9 9 8

2. DATE OF FILING
10/19/87

3. FREQUENCY OF ISSUE
Bi-monthly

3A. NO. OF ISSUES PUBLISHED ANNUALLY
6

3B. ANNUAL SUBSCRIPTION PRICE
$42.-U.S. $60.
$49.-CN/MX Int'l

4. COMPLETE MAILING ADDRESS OF KNOWN OFFICE OF PUBLICATION *(Street, City, County, State and ZIP+4 Code) (Not printers)*
34021 Granada, Suite B, Dana Point, CA 92629 ORANGE COUNTY
ALL MAIL TO: 24843 Del Prado, Suite 473, Dana Point, CA 92629-2852

5. COMPLETE MAILING ADDRESS OF THE HEADQUARTERS OF GENERAL BUSINESS OFFICES OF THE PUBLISHER *(Not printer)*
Same as item 4.

6. FULL NAMES AND COMPLETE MAILING ADDRESS OF PUBLISHER, EDITOR, AND MANAGING EDITOR *(This item MUST NOT be blank)*

PUBLISHER *(Name and Complete Mailing Address)*
William E. Jaenicke (same address as item 4.)

EDITOR *(Name and Complete Mailing Address)*
William E. Jaenicke " " " " " " " "

MANAGING EDITOR *(Name and Complete Mailing Address)*
William E. Jaenicke " " " " " " " "

7. OWNER *(If owned by a corporation, its name and address must be stated and also immediately thereunder the names and addresses of stockholders owning or holding 1 percent or more of total amount of stock. If not owned by a corporation, the names and addresses of the individual owners must be given. If owned by a partnership or other unincorporated firm, its name and address, as well as that of each individual must be given. If the publication is published by a nonprofit organization, its name and address must be stated.) (Item must be completed.)*

| FULL NAME | COMPLETE MAILING ADDRESS |
|---|---|
| The International Association of Cromemco Users, Inc. | Same as item 4. |
| Harry T. Garland, Ph.D. | 280 Bernardo Ave., Mt. View, CA 94039 |

8. KNOWN BONDHOLDERS, MORTGAGEES, AND OTHER SECURITY HOLDERS OWNING OR HOLDING 1 PERCENT OR MORE OF TOTAL AMOUNT OF BONDS, MORTGAGES OR OTHER SECURITIES *(If there are none, so state)*

| FULL NAME | COMPLETE MAILING ADDRESS |
|---|---|
| NONE | |

9. FOR COMPLETION BY NONPROFIT ORGANIZATIONS AUTHORIZED TO MAIL AT SPECIAL RATES *(Section 423.12 DMM only)*
The purpose, function, and nonprofit status of this organization and the exempt status for Federal income tax purposes *(Check one)*

(1) □ HAS NOT CHANGED DURING PRECEDING 12 MONTHS
(2) □ HAS CHANGED DURING PRECEDING 12 MONTHS
*(If changed, publisher must submit explanation of change with this statement.)*

| 10. EXTENT AND NATURE OF CIRCULATION *(See instructions on reverse side)* | AVERAGE NO. COPIES EACH ISSUE DURING PRECEDING 12 MONTHS | ACTUAL NO. COPIES OF SINGLE ISSUE PUBLISHED NEAREST TO FILING DATE |
|---|---|---|
| A. TOTAL NO. COPIES *(Net Press Run)* | 2,550 | 2,650 |
| B. PAID AND/OR REQUESTED CIRCULATION<br>1. Sales through dealers and carriers, street vendors and counter sales | 505 | 500 |
| 2. Mail Subscription *(Paid and/or requested)* | 1,095 | 1,001 |
| C. TOTAL PAID AND/OR REQUESTED CIRCULATION *(Sum of 10B1 and 10B2)* | 1,600 | 1,501 |
| D. FREE DISTRIBUTION BY MAIL, CARRIER OR OTHER MEANS SAMPLES, COMPLIMENTARY, AND OTHER FREE COPIES | 120 | 101 |
| E. TOTAL DISTRIBUTION *(Sum of C and D)* | 1,720 | 1,602 |
| F. COPIES NOT DISTRIBUTED<br>1. Office use, left over, unaccounted, spoiled after printing | 830 | 1,048 |
| 2. Return from News Agents | 0 | 0 |
| G. TOTAL *(Sum of E, F1 and 2—should equal net press run shown in A)* | 2,550 | 2,650 |

11. I certify that the statements made by me above are correct and complete
SIGNATURE AND TITLE OF EDITOR, PUBLISHER, BUSINESS MANAGER, OR OWNER

PS Form 3526, July 1984
*(See instruction on reverse)*

PHOTOTYPESET FOR QUALITY

# INPUT...

**Editor:**

RE the SOFT TOOLS column in *I/O NEWS Volume VI, Number 1*, I applaud (loudly) the section on "Rules of Tools," especially the references to using ANSI Standards. But I would like to point out that "FORTRAN 66" is not an ANSI Standard—see the third paragraph of the Foreword to ANSI X3.9-1978 (FORTRAN 77) which states that "American National Standard Basic FORTRAN, ANSI X3.10-1966, has been withdrawn." The "next" Fortran would seem to follow the same pattern—the most recent draft of "Fortran 8x" (June 1987) states in the Foreword that "the previous standard, commonly known as 'FORTRAN 77', is entirely contained within this standard, known as 'Fortran 8x'," and in the Introduction, paragraph 1.1 Purpose, states "this standard is an upward compatible extension to the preceding Fortran standard, X3.9-1978, informally referred to as FORTRAN 77."

*Herrick S. Lauson*
*Albuquerque, New Mexico*

**Editor:**

I tried to buy the dBASE II data base management software only to learn from Ashton-Tate and others (see below) that it is no longer available from anyone.

Ashton-Tate no longer supports the CP/M product. A software package that has been so widely distributed since the beginning of the 80's cannot have bugs that make working with it impossible. I regret that Ashton-Tate will no longer sell the package even though this attitude is very professional. One of the only reasonably priced systems for low-end computer users was dBASE II...

This is especially unfortunate for Cromix users. Other products like Wordtech Systems dBASE III Compiler or RDS Informix are specially tailored to a multiuser environment and perform much better, but are priced $1000 and up.

*Alberic Muller*
*Switzerland*

The following is an excerpt from a reply letter which Mr. Muller received from the CP/M Connection:

"With regards to dBASE II I can give you no good news. The company which produced this product, Ashton-Tate, has stopped all marketing of dBase II and only sells and supports dBase III, which runs under MS-DOS only."

"We have talked with them about the possibility of allowing us to market it, and they refuse to allow anyone to sell it because they will only offer support to people who bought dBase II before they stopped selling it..."

CP/M Connection
PO Box 236
McPherson, KS 67460
(316) 241-3100

**Editor's Reply:**

*It is indeed unfortunate that dBase II is no longer available to the CDOS or CP/M users. But from a marketing viewpoint, the reasons are all too clear: nobody makes Z80 computers anymore. Nevertheless, there are still many Z80 systems in use.*

*Perhaps that is where the solution lies—with the people still using the "low- end" systems. If any of you have any suggestions as to what can be done about this situation, please write* I/O NEWS *and share your ideas.*

**Bernoulli Box For Backup?**

I just got the news that Cromemco is selling a new backup device: the CTD-60. this new tape certainly is a big improvement on the CTD featureing DMA transfers and higher speed. When I read the price tag of $2995—excluding cables and ESDC, my enthusiasm dwindled.

I have been looking at IOMEGA ads (Bernoulli Box). IOMEGA makes 10 and 20 Mbytes disk drives with removable cartridge. It seems that they are now manufacturing them with the 5¼" form factor. Since they use the ST-506 interface, their new drive could probably be easily integrated in existing Cromemco systems. Most of us are using STDC anyway. If the drive works in a Cromemco system, this could be an alternative to the CTD-60. We would loose the larger capacity and the QIC-24 standard, but would retain the speed; all of this at a much lower cost.

I am wondering if anyone has ever tried this?

Sincerely,

*Alberic Muller*
*Switzerland*

*[The inquiries I made in regards to interfacing the IOMEGA drive to a Cromemco STDC controller failed to determine whether or not anyone has done this. If any of you have, or know of someone who has, please contact I/O NEWS. Ed.]*

**Dear Editor:**

Per our phone discussion the following benchmarks may be of interest to *I/O NEWS* readers [see table shown above]. These benchmarks are all real problems with the same code compiled and linked, between all machines. The linked program size was approximately 600 Kbytes.

Sincerely,
*Carl J. Wick*
*Gladstone, Michigan*

| Program Run On[2] | I/O Input | Analysis[1] | I/O Output |
|---|---|---|---|
| Univac 1100/80 [5] | 15 sec | 10–15 sec | 15 sec |
| IBM 4341 [3] | 30 sec | 1–2 min | 30 sec |
| CS 400 with XPU [4] | 30 sec | 4–6 mins | 30 sec |
| CS 400 with XXU [4] | 25 sec | 5–15 sec | 25 sec |
| IBM 3800 Series [3] | 25 sec | 7–20 sec | 25 sec |

The above times are all wall time, which is the only real value to an end user.

[1] Average time per block; does 1 to 150 blocks per run (there is no I/O operation in this section). Code all to ANSI 77 Fortran standards except for the IBM, which doesn't use all of the standards.

[2] Single user on all systems except Univac, which had about 10 users at the time of the benchmark.

[3] 4 megabyte machine in single user mode using IBM software. On the 3800 series machines I don't exact data.

[4] 2 meg machine with Unix (I think memory would help.

[5] Univac is a half meg word machine.

# OUTPUT...

## The State of the Association

As I write this we're well into 1988, and it seems appropriate to take a look at where the International Association of Cromemco Users has been, where it's going, and most importantly, where it stands today. And in so doing, we'll be able to reiterate just what the association is about.

When the association was born, late in 1980, "microcomputer" had yet to become a household word. Cromemco was among a handful of companies that were the first to manufacture and make them commercially available. The word travelled fast, and with the demand for hardware came the demand for software—the operating systems and programming languages that would bear the fruit of applications for end-users, all of which led to an immediate demand for information regarding Cromemco systems. It was this need for know-how that spawned the formation of the I.A.C.U.

And that has been its purpose ever since: to gather and disseminate information pertinent to Cromemco users. This publication, *I/O NEWS*, has been and remains dedicated entirely to that end.

Over 1,500 people received the first issue of *I/O NEWS* back in October of 1980. The membership of the I.A.C.U. continued to swell over the next three years, and more than doubled. During that time we saw a proliferation of new hardware and software products. Cromemco helped lead the way with faster processors, more memory, and a growing list of developmental and applications software. And due to the modular design of our Cromemco systems, we continued to upgrade and keep up with the latest technological advances.

Lisa Jaenicke

If there ever was a time when we as Cromemco computer users were part of the mainstream, that was it. The closest thing to a standard operating system back then was CP/M, and we had a better one in CDOS. But where we really stood above and apart from the rest of our computing colleagues was that we had something that they didn't, and that was the Cromix Operating System.

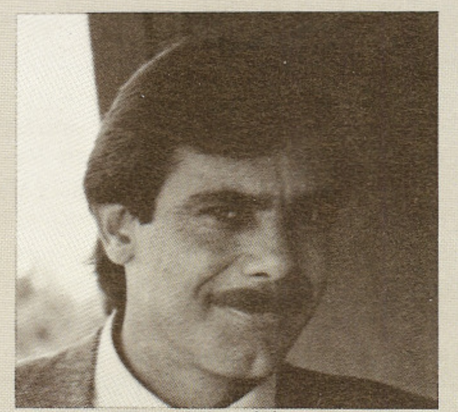
Cromix let us do things that others couldn't. We had multi-user capability on a microcomputer. We had multitasking. We had re-direction of input and output, and we had pipes. Our CP/M and CDOS software could still be run. And all the while the I.A.C.U. performed its function. We helped each other along, sharing that know-how which can be so hard to come by.

But by 1984 a new acronym had entered our collective vocabulary—PC—the "personal computer." That it happened so quickly can be attributed to the unprecedented marketing effort launched by IBM, the consumate marketeers. Along with the PC came MS-DOS. The main stream had parted.

As Cromemco users, we didn't feel it at first. Our hardware was superior—it could be expanded. We could be multiuser. And there was still a great deal of software development going on from which we did benefit directly. Commercial packages like Wordstar, dBase II, Multiplan, and SCADA, to name a few, were available under CDOS and Cromix. In addition there was a lot of new program development directed strictly to Cromemco systems, such as Procall, The Menu Generator, FastBack, the Gunn Simulator, and a host of others. And that helped keep us going.

But the sad fact is that MS-DOS soon became the primary target for the development of new software, despite the fact that it was functionally devoid of the capabilities inherent in Cromemco systems. And that hurt us. It marked the end of the expansion of the I.A.C.U. and the beginning of a decline.

So over the past few years the I.A.C.U. has had to tighten its belt. Although we've seen a good deal of attrition in the ranks we've also witnessed a great deal of devotion. Of the original 1711 members that joined during that first year, we can still count 346; 230 of the 450 who joined during the second Volume of *I/O NEWS*; 345 of the 756 from Volume III; 48 of the 84 from Volume IV, 21 of the 22 from Volume V, and all ten who joined with Volume VI. And that brings us to our present day size of 1000 members.



Bill Jaenicke

And now it appears that MS-DOS is going into its own decline; Unix beckons on the horizon. At present we are witnessing a resurgence in software development for 68000 Cromix, as evidenced by products such as dBIII Compiler, 68Kalc, Procall+, and the WP Word Processing system (see *New Products*), each of which have or will have their Unix counterparts.

So where do we stand as an association?

The answer to that is: intact, but with fewer members. What we have is a Catch-22 situation. In order to grow we need to increase our advertising revenues and increase our membership base. But Cromemco's sales are becoming more and more focused on specialized OEM markets; their sales no longer translate into IACU memberships. And that's the catch: a small circulation makes for a higher advertising C.P.M. (cost per thousand), which is the measure by which advertisers spend their marketing dollars.

What can be done to reverse this trend?

At this end we can cut out some of the "sizzle" and keep the editorial "steak," which we've done this issue by abandoning the four-color printing. We can

*Continued on page 30*

# Trapping of Bad Blocks On Cromix Machines

*by Steve Jones & Colin Liebenrood*

**Editor's Note:**

*The following article appears courtesy of CUG UK as it was published in the* CUG Newsletter, *No. 17, July 1987.*

Trapping Of Bad Blocks On Cromix Machines *by Steve Jones (Museum Documentation Association) and Colin Liebenrood (National Maritime Museum)*

### 1.0 Introduction

It is not uncommon to find that some parts of the magnetic medium used in a computer disk are unable to retain the information written upon them. The disk is tested during the manufacture to see if it is adequate for use. On hard disks, due to the difficulty of ensuring that the whole of the medium is usable, provision is made to allocate alternate tracks so as to avoid bad portions of the disk. This allocation can only be made when the whole disk is formatted.

During subsequent use, additional bad patches may appear, which are reported as disk errors. Whilst these can be covered by an additional alternate track allocation, this involves unloading all the files, re-formatting and re-loading — a lengthy process with a large disk. An alternative is to "trap" the bad block in an unwanted file. A technique for doing this on Cromemco systems running Cromix has been developed by Steve Jones of MDA. This technique is described below.

### 2.0 Procedure

**Not for the inexperienced or non-technical user!** Have you got a back-up in case it goes wrong?

The technique is described using a partition of a SMD hard disk as the device with the bad block. However, the technique will work for all Cromix disks.

### 2.1 Initial Steps

The first step is to record the disk error and identify its effect on operation.

(A) Write down the error message, which identifies the device and the block involved.

(B) Identify the device-name and its root directory from the device number. If necessary, use l /dev and look for the name corresponding to the device number in the error-message. Then use the mount command to identify its root directory.

(C) Use icheck -b [block-number] [device-name] to find the current use of the block, either in a file or on the free list.

(D) If it is in a file, then the inode number is given. Use ncheck -i [block-number] [directory] to get the pathname, where [directory] is the root directory of the device. Use the move command to transfer the file to a directory reserved for the purpose, such as bad, which should be in the root directory of the same device. Further action can then be taken when convenient. Note that the bad directory must be on the SAME device, so that the move command simply alters directory entries.

(E) If it is on the free list, then proceed to the next stage straight away, to prevent the block from being re-used. See para 2.3 below.

### 2.2 An Example of the Initial Steps

The following example illustrates these initial steps:

A.
```
Disk Verify error 4:0 Blk 38615 cyl 6B Surf 12 Sec 11
Stat 904
```

B.
```
% l /dev
Directory: /dev
...
 1:4 B 1 sfda
 4:0 B 1 smd
 4:1 B 1 smdpl
...
% mount
 /dev/root          /
 /dev/sfda          /a
 /dev/smdpl         /hd
% root
 /dev/smd
```

The affected device is smd — the Cromix root device — with root directory /.

C.
```
% priv
# icheck -b 38615 smd
 --- block in file, inode 876 ---
```

D.
```
# ncheck -i 876 /
 876 l /mrf/mrf6a.bdf
# move -v /mrf/mrf6a.bdf /bad/block38615
/bad/block38615
#
```

### 2.3 Procedure to Trap One Block

This must be carried out with NO other users on the machine, and in privileged mode. The steps are:

A. Create a one-block file in the bad directory of the affected device.

B. Obtain the inode number of the new file.

C. Use idump to get the disk address of its inode.

D. Delete the file containing the bad block (if any).

E. Patch the inode of the new file to make the bad block its data block.

F. Run check -s to correct the disk structure. Check that the correct block is in the new bad_block file. Make the file inaccessable to prevent future read errors.

### 2.4 Example of Bad-block Trapping

A. In the bad directory of the affected device, create a one block file. For example:

```
# d /bad
# input > bb38615
this is a bad block
#
```

B. Use l -i to get its inode number:

```
# l -i bb38615
 726 1 bb38615
#
```

C. Use idump to display the inode of the new file.

```
# idump /dev/smd 726
Inode 726, byte 019280H
0000: 00 00 00 00 00 00 00 80 01 00 00 00 00 0B 02 D6
0010: 02 0D 00 00 00 00 00 01 55 07 11 10 18 0F 55 07
0020: 11 10 18 1A 55 07 13 08 2E 13 00 00 00 00 00 00
0030: 00 00 FD 45 00 00 00 00 00 00 00 00 00 00 00 00
0040: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0050: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Note the hexadecimal number of the start of the inode.

The block number used by this file is in the 4 bytes starting at displacement 30: in the inode. (00 00 FD 45). (For a detailed description of the contents of the inode, see the Cromix system Manuals.)

D. Delete the original file (if any) containing the bad block.

E. Find the hexadecimal equivalent of the bad block number. If you don't have a handy calculator or a program to do this, then use dump with the begin option at the required number:

```
# dump -b 38615 /dev/smd
0096D7: 00 00 00 00 ...
```

The required hexadecimal equivalent is at the start of the displayed line.

Use patch [device-name] to alter the block number in this inode:

block number in inode starts at 019280h + 30h = 192B0h; value to be 38615 = 000096D7h

```
# patch /dev/smd
 > s192b0
 00192B0: 00 0
 00192B1: 00 0
 00192B2: FD 96
 00192B3: 45 D7
 00192B4: 00 .
 > e
#
```

F. Did you do it right? Use idump again to check! Run check -s [device-name] to correct the disk structure. After the resulting re-boot, use icheck -b (as in step 2.1 C) to verify that the bad block is in the correct file in the bad__block directory.

Finally, use the access command to alter the protection of the new file so that it cannot be read or written:

```
# access .. bb38615
# l -l
Directory: /bad
11    1 ---- ---- ---- system Jul-17 16:24 bb38615
```

The system can then be returned to normal use.

### More On Recovering Hard Disks *by Adrian Pickering*

Having once spent 8 hours or so helping a colleague recover his filestore (on a Rodime R0204) the moral of this tale was clear. You MUST have a paper copy of the output of the Cromix program diskinfo for each of the drives you use in your machine. This should be stuck to the drive and/or in the machine log book. The "achilles heel" of ST506 drives as used in Cromix is the disk configuration information recorded on cylinder 0. If this goes, which is not unlikely as the heads "park" around here, then it is very difficult to recover without having a diskinfo printout. If you haven't got diskinfo — get it now!

The most common fault...are contacts! Cromemco learned this a while back when they gave up fully socketting all the ICs on their cards and tightening up the backplanes. However, not all ICs are soldered in and problems can still arise.

Twice I have had a machine appear to work fine for a single user and just lock up the moment more processes are added. This was due to gradual tarnishing of the legs of the 5501 on the FDC. The symptom is easy to spot in that ps -al yielded the usual response but with processes never taking any CPU time! The 5501 was obviously not generating the time-slicing interrupts required.

I don't know what sort of tin some of these IC manufacturers use on the legs of some ICs but it seems to get mixed up with monsoon mud. If you are anxious about removing the ICs and giving the underside of their legs a good scrape with a scalpel, then just prise one side up and press it down again in its socket a side at a time and repeat a few times. This should sufficiently scrape the tarnish off for another few months of operation.

Of course, there are other ways to ensure that contacts don't give problems:

- don't touch any part of the internals of the machine; leave it alone. (I wish I could do this!)
- don't switch the machine off. This avoids thermal cycling of the internals as well as reducing stress on hard-worked components such as regulators and hard-disk motor drivers and bearings. All the component failures I have had in the last year have occurred on power up.
- don't switch the machine on (most efficacious).

# FORTRAN-88

## The Next Fortran Standard?

*by Herrick S. Lauson*

FORTRAN 77 (as defined in ANSI X3.9-1978, American National Standard Programming Language FORTRAN) was approved by ANSI on 3 April 1978 and accepted by the Departments of the Army, Navy and Air Force on 15 November 1978. The Fortran Committee (X3J3) continued, with an original target date of 1983 for the next Fortran Standard. This slipped several times for various reasons and in March 1986 the first ballot of the Committee to forward the draft standard to X3 for release for public comment failed. The language was reduced, in December 1986 the second ballot passed, and, after comment processing and editorial revision, was accepted by X3 in August 1987. As of this writing, I do not know when the four-month public review period is to start. However, speculation among committee members seemed to agree on a date of 1 October 1987 for availability of the draft standard (at a cost of near $100). Copies may be obtained from Global Engineering Documents, Inc. by calling 1-800-854-7179 (West Coast) or 1-800-248-0084 (East Coast).

A major reason for the failure of the ballot in March 1986 was the size of the language. It's still too big for some people and others want more features! Some of the features that were removed in 1986 are bit data type, arrays of arrays, vector-valued subscripts, nesting of internal procedures and condition handling. Many people would like a pointer data type and discussion continues on this but it probably won't be in the language until the next revision (already being referred to as Fortran 9x). A great deal of effort has been put into ensuring that FORTRAN 77 is included in the current revision but there probably will be a few minor exceptions.

**Array Processing** is the largest revision. 28 (of 59) new intrinsic functions are related to array processing. Among these are:

```
DOTPRODUCT(VECTOR_A,VECTOR_B)
MATMUL(MATRIX_A,MATRIX_B)
TRANSPOSE(MATRIX)
```

which are obvious, and others refer to array reduction, inquiry, construction, manipulation and geometric location. Users may also specify rectangular subarrays (via implied DO loops for indices) to be used as arrays and skew subarrays (via an IDENTIFY mechanism for an ALIAS name) to be used as rank-one arrays (great for accessing diagonals!). Arrays may also be allocated and deallocated, giving users a form of dynamic memory control. One could almost write a book on array processing alone!

**Derived Date Types** are akin to Pascal's enumerated types, allowing the user to define "collections" of data of different types. Example:

```
TYPE PERSON
INTEGER AGE
CHARACTER (LEN = 50) NAME
END TYPE PERSON
```

accessed as PERSON%AGE and PERSON%NAME. The IMPLICIT NONE type specifier is also available — great for those who believe, as I do, in explicit typing. Type specifications may also include attributes such as PARAMETER, DATA, PUBLIC or PRIVATE access, SAVE, ALIAS and RANGE. Examples:

```
REAL, PARAMETER :: ONE = 1.0
INTEGER, DATA :: NEXT = 1
REAL, PRIVATE :: X,Y,Z
```

Modules, akin to ADA packages, are collections of data, type definitions, and procedure definitions and are accessed via the USE module_name statement. This feature will provide a replacement for COMMON, BLOCK DATA and the non-standard INCLUDE extensions.

**Enhanced Procedure Features** allow internal procedures, optional arguments (which may be queried by the PRESENT intrinsic function), specification as RECURSIVE, argument specification as IN, OUT, or INOUT (via the INTENT attribute), and the use of keyword calls such as:

```
CALL PROC(keyword = actual_arg).
```

**New Control Features** include the DO-END DO and the CASE constructs. DO, IF, and CASE constructs may be named. DO loops also have the (N) TIMES, EXIT (branch just past the end of the [named] loop) and CYCLE (branch to the end of the [named] loop) features. Example:

```
OUTER: DO                   !Unlimited DO, named OUTER
          ...
          DO (N) TIMES      !N executions
          ...
          DO I = 1,N        !I = 1,2,...,N
          ...
           IF (...) EXIT OUTER !Possibly exit outer loop
          END DO
          IF (...) CYCLE    !Possible skip to end of the loop
          ...
          END DO
END DO OUTER
```

**Generalized Precision** allows specification of (minimum) precision in the form of, for example, REAL(10,99) (10 significant decimals and range of at least [10**-99,10**99] — single precision on a Cray). There are also 9 intrinsic functions allowing user access to the specifications of the numeric model and 8 floating-point manipulation functions.

**Input/Output** has many minor enhancements (especially for OPEN and INQUIRE statements) and one major new feature — NAMELIST (an extension in many implementations of FORTRAN 77).

**Enhanced Character Set** now includes ! " % & ; < > ? [ ] _ and lower case (if the processor recognizes it). The underscore (_) is alphanumeric and no distinction is made between upper and lower case (except in character strings).

**Variable Names** may contain up to 31 alphanumeric characters (still starting with a letter).

**Alternate Operators** are allowed — .LT., .GT., .LE., .GE., .EQ., and .NE. now have the alternate "spellings" < , > , <=, >= , = = , and <>.

**Free Source Form** uses ! (anywhere on the line) as the beginning of a comment, uses & (as last non-blank character on a line) to flag continuation to the next line, allows multiple statements/line (separated by ; ), 132 characters/line, 2640 characters/statement. The standard fixed source form is still available, of course, but the two forms may not be mixed in the same program unit. The means for specifying the source form are not defined in the standard. (Note that a C or * in column 1 may be used only in the fixed source form and the & continuation may be used only in the fixed source form.)

**Other New Intrinsics** include a number of useful character functions (remove trailing blanks, ASCII analogs of CHAR and ICHAR, optional backward scanning for some functions, etc.), a random number generator, a date and time function, and a system clock function.

**Deleted** features are non-existent at the present time.

**Obsolescent** features are those recommended to be avoided and to be considered for possible deletion in the next revision. These are arithmetic IF, real and double precision DO control variables, shared DO terminations, branching to an END IF from outside its IF block, alternate returns, PAUSE, ASSIGN and assigned GO TO, and assigned FORMAT specifiers.

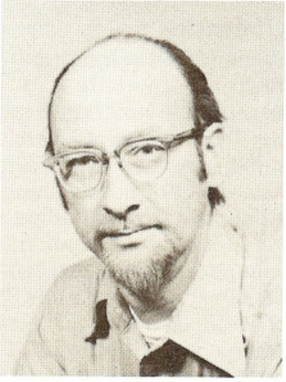
**Deprecated** features are those expected to become obsolete because of new features in this revision. Some of these are BLOCK DATA, COMMON, EQUIVALENCE, ENTRY, computed GO TO, statement functions, specific names for generic functions, DIMENSION and DOUBLE PRECISION.

**When** can we expect to get all this neat stuff? — maybe early 1989. The Fortran committee must answer all comments received during the public review period. This process could well take more than a year (it took 18 months for FORTRAN 77).

**About the Author:**

Herrick S. (Hank) Lauson *is the owner of:*

hazel graphics
7711 Euclid Ave, NE
Albuquerque, NM 87110-4803



*and is currently working on FORTRAN-77 implementations of ANSI graphics standards. A Fortran programmer since 1960 (mainly for large hydrodynamic codes with emphasis on graphics output), he recently retired after 8½ years at Los Alamos Scientific Laboratory and 20 years with Sandia National Laboratories, where his last assignments were writing device drivers for the Sandia Virtual Device Interface and evaluating ANSI GKS (Graphical Kernel System) packages. He is an active member of the ACM and shortly after retirement was appointed an alternate member of the ANSI Fortran Committee. He has owned an evolving Cromemco system (Z-2 based) since early 1983.*

# Using the Cromemco SDI Color Graphics Interface And SGS Graphics Software With Digital Research PL/I-80 Programs

*by Robert J. Diersing*

### Introduction

Having been a long time user of the PL/I language on IBM mainframe computers, I was very pleased when Digital Research announced its implementation of the PL/I-80 compiler. I have used all versions of the compiler on CDOS and Cromix systems with complete success.

As a result of my teaching assignments, work as an independent consultant, and doing a considerable amount of programming as a hobby, I come into contact with almost every piece of computing hardware Cromemco has ever made. Since I have access to systems with the SDI graphics hardware both at home and at the university, I have often wanted to use the SDI/SGS combination with my PL/I-80 programs. At one time I considered writing a new graphics library that would interface with PL/I-80 and the SDI hardware. It has turned out that this effort was unnecessary.

While working on an entirely different project I discovered that the SGS graphics .REL files could be linked with PL/I-80 modules with the linkage editor as supplied by Digital Research. This may be common knowledge to some readers but it was news to me. A few preliminary tests confirmed that it would only be necessary to write a small interface routine to resolve the differences between the PL/I-80 and SGS parameter passing conventions. The SGS routines could then be used with PL/I-80 as documented in the SGS literature. This article describes the details of the required interface and two alternatives for coding the PL/I-80 programs.

### Parameter Passing and Interface Requirements

Before describing the interface in detail we will examine the differences in parameter passing conventions between PL/I-80 and a Cromemco-supplied language, say Fortran. Figure 1 shows the parameter passing conventions for PL/I-80. The address fields are arbitrary and there is no difference in format, other than the length of the parameter block, regardless of the number of parameters passed.

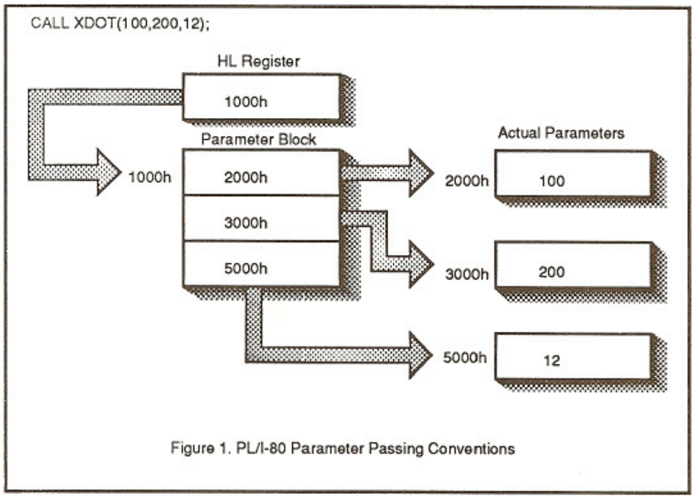


Figure 1. PL/I-80 Parameter Passing Conventions

Let us assume a typical SGS graphics call in PL/I-80:

```
CALL XDOT (100,200,12);
```

There are two possible parameter passing conventions from Cromemco-supplied languages. One method is used when there are three or fewer parameters while a different one is used when there are more than three parameters. Figure 2 shows an example of each case.

When using SGS routines with Cromemco C, Z-80 assembler, or Fortran programs and there are three or fewer parameters, the HL, DE, and BC registers point to the first, second, and third parameters respectively. If four or more parameters are passed, HL and DE point to the first two parameters while BC points to a block of 2-byte pointers. The first word in the block points to the third parameter, the second word to the fourth parameter, and so forth. The implementation of this block of pointer words is identical to the procedure PL/I-80 uses no matter how many parameters happen to be passed.

One of the functions of the interface will have to be rearranging the PL/I-80 parameter conventions so that they match SGS conventions. Another problem that arises is that due to the differences in parameter passing conventions, the SGS entry point cannot be called directly. This is to say that XDOT, for example, cannot have the attributes Entry and External. The PL/I-80 program will have to call entry points to the interface routine that have different names from the actual SGS entry point names.

Unfortunately, the differences in parameter passing conventions, and thus not being able to call SGS entry names directly, are not the only problems. The rules for coding PL/I-80 procedure Call statements (regardless of whether they are internal or external procedures) require that the number and types of variables in a Call statement be specified exactly in the declaration of the procedure entry point. Here is an example.

Suppose we need to call the SGS entry point XDOT. We know we have to call an interface routine named different than XDOT. Furthermore, three integer parameters are required — one for the X coordinate, one for the Y coordinate, and one for the dot color. Lets suppose we create an entry point in our interface routine that handles all calls with three parameters destined for SGS. We will call it SGS3P. Of course SGS3P will need to know which of the SGS entry points to call since many SGS entry points have three parameters. The following are the PL/I-80 statements that will be used.

```
Declare SGS3P         Entry (Fixed Binary (15),
                      Fixed Binary (15), Fixed Binary (15),
                      Fixed Binary (15)) External;
Declare XDOT          Fixed Binary (15) Static Initial ( 2);
        CALL SGS3P    (XDOT, 100, 200, 12);
```

Of the above three statements, the first defines the entry point in the PLISGS interface that is to be called. It will be called for all cases where three fixed binary (15) integer parameters must be processed. The additional parameter XDOT is also passed to SGS3P. It is the value of the variable XDOT that will allow SGS3P to finally pass control to the actual XDOT

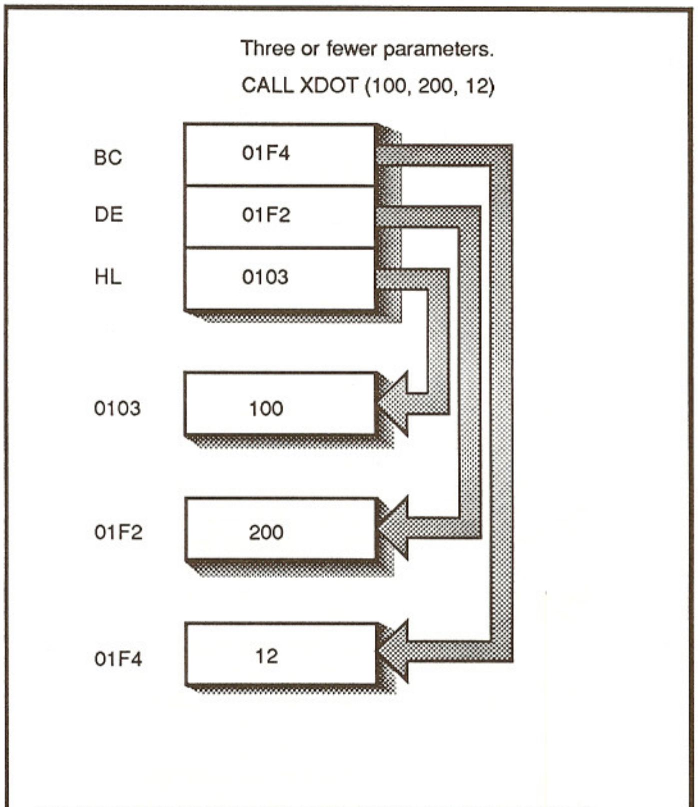


**Figure 2a: Cromemco Parameter Passing Convention 1**

entry point in the SGS graphics routines. The parameters 100, 200, and 12 will be passed according to the rules for the SGS package. They could just as well have been variables rather than self-defining terms.

This scheme requires that an SGSnP entry point be declared for each possible parameter configuration but there are not too many variations. Once coded these statements are placed in a file called PLISGSEP.PLI from which they can be %INCLUDEd whenever they are needed. A complete list of possibilities is shown below along with an explanation of their use.

```
/* No parameters passed to SGS */
Declare SGS0P          Entry (Fixed Binary (15)) External;

/* One integer parameter passed to SGS */
Declare SGS1P          Entry (Fixed Binary (15),
                       Fixed Binary (15)) External;

/* One integer array parameter passed to SGS */
Declare SGS1PA         Entry (Fixed Binary (15),
                       (64) Fixed Binary (15)) External;

/* Two integer parameters passed to SGS */
Declare SGS2P          Entry (Fixed Binary (15),
                       Fixed Binary (15), Fixed Binary (15))
                       External;

/* One integer scalar and one integer array passed to SGS */
Declare SGS2PA         Entry (Fixed Binary (15),
                       Fixed Binary (15),
                       (64) Fixed Binary (15)) External;

/* Three integer scalar parameters passed to SGS */
Declare SGS3P          Entry (Fixed Binary (15),
                       Fixed Binary (15), Fixed Binary (15)
                       Fixed Binary (15)) External;
```

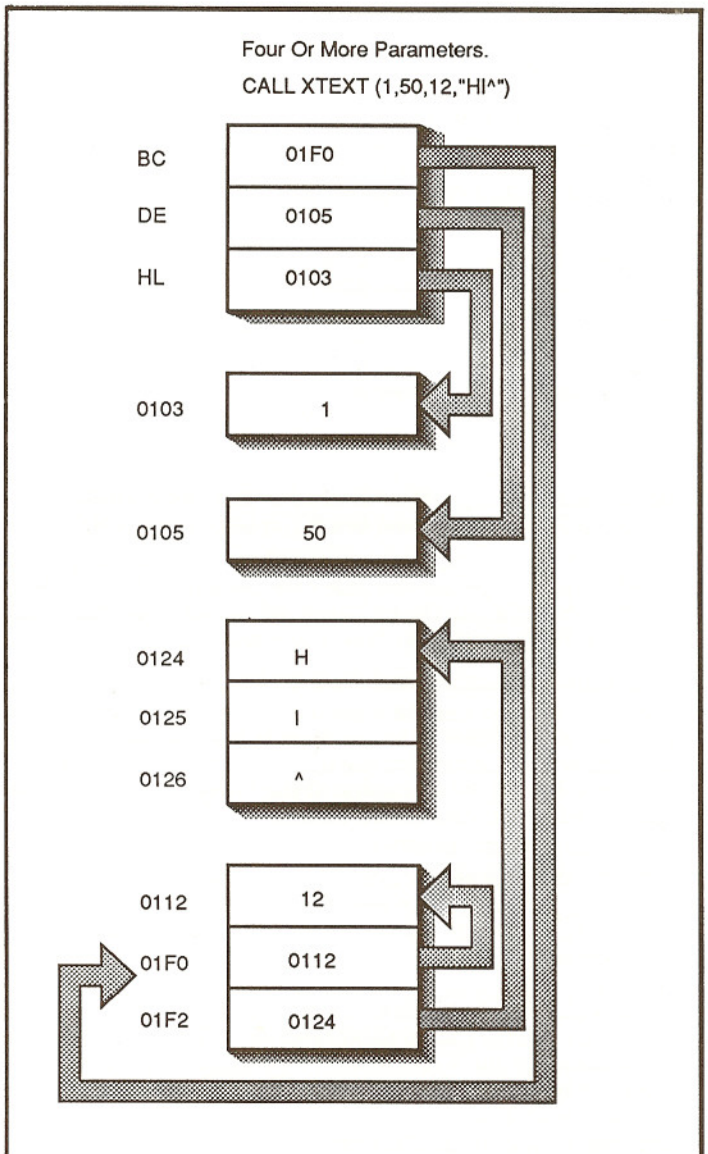


**Figure 2b: Cromemco Parameter Passing Convention 2**

```
/* Two integer scalar and one character string to SGS */
Declare SGS3PM         Entry (Fixed Binary (15),
                       Fixed Binary (15), Fixed Binary (15)
                       Char (254) Varying) External;

/* Four integer scalar parameters passed to SGS */
Declare SGS4P          Entry (Fixed Binary (15),
                       Fixed Binary (15), Fixed Binary (15))
                       Fixed Binary (15), Fixed Binary (15))
                       External;

/* Three integer scalar and one character string to SGS */
Declare SGS3PM         Entry (Fixed Binary (15),
                       Fixed Binary (15), Fixed Binary (15)
                       Fixed Binary (15), Char (254) Varying)
                       External;

/* Five integer scalar parameters passed to SGS */
Declare SGS5P          Entry (Fixed Binary (15),
                       Fixed Binary (15), Fixed Binary (15)
                       Fixed Binary (15), Fixed Binary (15)
                       Fixed Binary (15)) External;
```

*Continued on page 12*

# Graphics

*Continued from page 11*

In addition to the above list of entry point names, a list of command variables will also be needed. A partial listing is given below. The complete set of these variables are placed in the file **SGSCMD.PLI** from which they can be **%INCLUDEd** as needed.

```
Declare XLINE        Fixed Binary (15) Static Initial ( 0);
Declare HXLINE       Fixed Binary (15) Static Initial ( 1);
Declare XDOT         Fixed Binary (15) Static Initial ( 2);
Declare HXDOT        Fixed Binary (15) Static Initial ( 3);
Declare XAREA        Fixed Binary (15) Static Initial ( 4);
Declare HXAREA       Fixed Binary (15) Static Initial ( 5);
 . . . . .
          . . . . . . . . . . .
                     . . . . . . . . . . . . . . . . .  . . . . . . . . . . . . . . . . .
Declare XORON        Fixed Binary (15) Static Initial (108);
Declare XOROFF       Fixed Binary (15) Static Initial (109);
```

Notice that the values given to these variables match the numbering scheme for the SGS calls established by Cromemco. This should allow the interface to be easily modified as new calls are implemented in future software releases.

### Interface Operations and PL/I-80 Program Coding

With the static variables required in the PL/I-80 program defined, we can now turn our attention to the actual PL/I-80 executable statements and the operation of the Z-80 assembler interface. During the discussion of the interface operation, consider the PL/I-80 call shown below.

```
        CALL SGS4PM (XTEXT, 1, 50, 12, 'HI∧');
```

Notice that the above call is very similar in format to other languages with which SGS routines can be used. The entry point **SGS4PM** exists in the interface and is called because there are four SGS parameters and one of them is a character string. The command code parameter is not counted because it is used by the interface to determine which SGS entry point to call and is not a parameter passed to SGS routine **XTEXT**. The other four parameters are the X coordinate, Y coordinate, color code, and the message.

A note with respect to the end-of-string delimiter "∧" is in order here. Due to sematic rules of PL/I-80, the usage as shown here is not allowed. In PL/I-80, the use of the "∧" character causes the following character in a literal character to be generated with certain bits turned off. Thus, a character such as ESCAPE can be generated by ∧[. This poses no problem though, because the SGS package includes the **ESTRNG** call to change the end-of-string delimiter to any desired character. I frequently use the "|" character for this purpose.

No matter which entry point to the **PLISGS** interface routine is used, the first order of business is to store the value of the first parameter. This is the command code parameter and its value will be used later to compute the displacement into the branch vector to find the proper SGS entry address. The following code is required to fetch the command code value. Remember that the HL register is pointing to the list of parameter addresses.

```
GETEP                         ;extract the entry point number
        LD      E,(HL)        ;get low order byte addr p1
        INC     HL            ;point to next byte of addr p1
        LD      D,(HL)        ;get high order byte addr p1
        INC     HL            ;point to next parm addr
        LD      A,(DE)        ;get p1 low order byte
        LD      (EP),A        ;save p1 low order byte
        INC     DE            ;point to high order byte
        LD      A,(DE)        ;get p1 high order byte
        LD      (EP+1),A      ;save p1 high order byte
```

At this point the command code value has been saved in the variable **EP**. Since the call was to **SGS4PM**, the code for that routine follows. This routine constructs the parameter list as required by the **XTEXT** entry point in the SGS from the parameter list received from PL/I-80. Register **HL** was left pointing to the address of the second parameter (the first actual SGS parameter) by the **GETEP** routine.

```
SGS4PM                        ;four parameters, one is char str
        LD      E,(HL)        ;get low order byte addr SGS p1
        INC     HL            ;point to high order byte
        LD      D,(HL)        ;get high order byte addr SGS p1
        INC     HL            ;point to next parameter
        PUSH    DE            ;save address SGS p1
        LD      E,(HL)        ;get low order byte addr SGS p2
        INC     HL            ;point to high order byte
        LD      D,(HL)        ;get high order byte addr SGS p2
        INC     HL            ;point next byte
        PUSH    DE            ;save address SGS p2
        PUSH    HL            ;save ptr to rest of list for p3
        INC     HL            ;point to next byte
        INC     HL            ;point to next byte
        LD      E,(HL)        ;get low order byte addr SGS p4
        INC     HL            ;point to high order byte
        LD      D,(HL)        ;get high order byte addr SGS p4
        INC     DE            ;skip over length in string
        LD      (HL),D        ;save high order byte addr SGS p4
        DEC     HL            ;point to low order byte
        LD      (HL),E        ;save low order byte addr SGS p4
        POP     BC            ;get back addr SGS p3, p4
        POP     DE            ;get back addr SGS p2
        POP     HL            ;get back addr SGS p1
        JP      JPSGS         ;go to get SGS entry point
```

At this point the parameter list as created by PL/I-80 has been modified as required by SGS. Note that one of the tricks in this routine is to skip over the string length descriptor at the beginning of the character string. The length is not needed by SGS since it searches for the end-of-string delimiter as defined by **ESTRNG**. Remember it must be something other than "∧". The only work that remains is the conversion of the SGS entry point number into an actual SGS entry point address. This is done with the routine below. Note that the SGS routine is reached via a **JP** instruction so that the **RET** instruction in the SGS routine will return directly to the PL/I-80 program.

```
JPSGS                         ;jump to proper SGS routine
        PUSH    HL            ;save HL
        PUSH    DE            ;save DE
        PUSH    BC            ;save BC
        LD      HL,(EP)       ;get previously stored entry number
        ADD     HL,HL         ;allow for word length entries
        LD      DE,JPVECTOR   ;get address of jump vector
        ADD     HL,DE         ;compute displacement
        LD      A,(HL)        ;get low order byte of SGS entry
        LD      (EP),A        ;save low order byte of SGS entry
        INC     HL            ;point to high order byte
        LD      A,(HL)        ;get high order byte of SGS entry
        LD      (EP+1),A      ;save high order byte of SGS entry
        POP     BC            ;restore BC
        POP     DE            ;restore DE
        POP     HL            ;restore HL
        LD      IX,(EP)       ;get SGS entry address
        JP      (IX)          ;jump to proper SGS routine
```

### An Alternate Approach to PL/I-80 Program Coding

The previous section presented one method of calling the PL/I-80 interface to SGS routines and the operation of a portion of the interface routine itself. That method required knowledge of the proper SGS interface routine entry point to match the number and type of parameters passed. I have not found that this is very troublesome but some programmers may prefer calling routines with exactly the same names as those found in the SGS package. In other words, a programmer may prefer to code:

```
CALL XTEXT (1,50,12, 'HI|');
```

in a PL/I-80 program rather than:

```
CALL SGS4PM (XTEXT, 1, 50, 12, 'HI|');
```

as suggested in the previous section. This extra convenience can be had at the expense of program and symbol table space used when the code to support this method is added.

In order to implement a scheme where the programmer calls routine names which are exactly the same as those used in SGS, there must be internal PL/I-80 procedures with the required names. These procedures will be very short and contain only the declarations of required variables and a call to the correct PL/I-80-SGS entry point described previously. Because these exact names are internal to PL/I-80 they will not cause problems later at link edit time.

This inclusion of many short procedures is wasteful of symbol table and stack space during compilation as well as program memory at turn time. The programmer must decide if the extra convenience is worth the system resources consumed. If only a small subset of the SGS entry names are needed then the procedure would probably be acceptable.

Some experimentation has been done with this method of using the PL/I-80-SGS interface. When it is used, the declarations for each entry point configuration are still needed. These were described earlier and are placed in a file called PLISGSEP.PLI which can be %INCLUDEd as needed. The list of commands stored in the file SGSCMD.PLI is not needed, however. If the programmer wanted to use the statement:

```
        CALL XTEXT    (1, 50, 12, 'HI|');
```

then the following interface procedure would have to exist in the PL/I-80 program.

```
XTEXT: PROCEDURE (X, Y, C, Message);
Declare  (X,Y,C)         Fixed Binary (15);
Declare  Message         Char (254) Varying;
         CALL SGS4PM     (12, X, Y, C, Message);
END XTEXT;
```

It can be seen, of course, that this is exactly the same operation as having called SGS4PM directly but it does allow the use of the XTEXT routine name within the PL/I-80 program. This procedure may be helpful when converting programs from other languages such as Fortran.

If this method of operation is chosen it will be useful to code all the required procedures and save them in files that can be %INCLUDEd in the PL/I-80 program. Furthermore, the procedures can be divided into logical groups so that only likely combinations appear together thus conserving resources during compilation and at run time. A suggested grouping of the SGS routine names is given below.

SGSHOUSE.PLI — Initialization and housekeeping functions.
AFILL, CLIP, ERASE, FILINT, FORE, GRAFIX, INIT, INIT1, PAGE, SCROFF, SCRON, UNCLIP, UNSCAL, XORON, XOROFF, COLR, DISP, PSPACE, RES, CURSOR, HCURSR, BFILL, DEFC, DEFCLR, ERABOX, HSCALE, RESBOX, SCALE, ESTRNG, LOAD, and SET.

SGSWNDOW.PLI — Animation and windowing functions.
ANIM, ANIMAT, WENAB, AINIT, WDISAB, WEXIT, WINIT, WORKON, WCLOSE, and WOPEN.

SGSMRESI.PLI — Medium resolution implicit functions.
DOT, AREA, FSEG, LINE, RLINE, READ, ELPS, FELPS, TEXT, and UTEXT.

SGSMRESX.PLI — Medium resolution explicit functions.
XFILL, XFPOLY, XDOT, XREAD. XCIRC, XFCIR, XAREA, XLINE, and XTEXT.

SGSHRESI.PLI — High resolution implicit functions.
HAREA, HDOT, HLINE, HRLINE, HREAD, HELPS, HFELPS, HTEXT, and HUTEXT.

SGSHRESX.PLI — High resolution explicit functions.
HXPOLY, HXDOT, HXREAD, HXCIRC, HXFCIR, HXAREA, HXLINE, and HXTEXT.

SGSMRESM.PLI — Medium resolution minimum system.
GRAFIX, INIT, INTI1, PAGE, COLR, DISP, CURSOR, DEFCLR, ESTRNG, WORKON, DOT, XDOT, INC, XLINE, READ, XREAD, TEXT, and XTEXT.

SGSHWARE.PLI — Hardware control.
WAITGH, WAITOD, WAITVG, and WAITVS.

### Extent of Test on the PL/I-80 to SGS Interface

The interface described in this article has been tested in several ways. First, most all of the example programs written in Fortran and supplied with the SGS software package have been converted to PL/I-80 and successfully tested. Second, many of the examples supplied in the SGS documentation manual have also been converted and tested. Finally, a number of programs typically assigned in a first computer graphics course have been coded and tested. These include programs to accomplish the standard two-dimensional functions of translation, scaling, reflection, rotation, line plotting, line clipping, and polygon clipping. Programs have also been written to handle three-dimensional problems.

I believe the conversion of the example programs for both the SGS package and manual have provided the best tests, however. This is because they exercise many of the features of the SGS system that are not typically used in the college assignments mentioned previously.

### Conclusion

I have been particularly pleased to find out that the SGS package is so easy to interface to PL/I-80 programs. Whether or not this continues to be true with future versions of the SGS software remains to be seen. The software described here has been tested with SGS release 6.

### About the Author:

Robert J. Diersing *is an Associate Professor of Computer Science at Corpus Christi State University, 6300 Ocean Drive, Corpus Christi, Texas 78412.*

# IT'S HERE!!

# dBIII Compiler

## for UniPlus + System V

Software Standards is now shipping dBIIICompiler from WordTech Systems for Cromemco's UniPlus System V machines. dBIIICompiler is a powerful dBase III compiler as well as a full-function program development system. No other software or product (such as dBase III) is required to use dBIIICompiler although it is compatible with dBase II and dBase III files from any other machine. With dBIIICompiler under UniPlus System V, you can quickly develop or port anything from small utilities to giant multi-user data processing systems. In addition to all the features found in our Cromix-Plus version, including multi-user record level locking, terminal independence and dBase II data file compatability, our new UniPlus version generates memory-efficient **SHARED CODE COMPILED PROGRAMS** and is 100% source code compatible with our Cromix-Plus version. Our latest release also incorporates **MULTIPLE PRINTER** support and complete control over the spooler. Suggested list price is $995.00 and there are no royalties for distributing compiler programs.

## RealWorld Accounting Packages

RealWorld is a full-featured accounting system which includes A/P, A/R, G/L, Payroll, Inventory, Order Entry, Billing, and Sales Analysis. All modules are fully integrated and the system is multi-RealWorld is available on both Z80 and 68000 Cromix as well as UniPlus System V. Suggested list price is $695.00 for Z80 versions and $795.00 for 68000 Cromix and UniPlus Versions.

---

Contact your Cromemco dealer for more information.
Dealers contact Software Standards for Discount Schedules, Literature and Demo Packages.

---

Software
Standards, Inc.

P.O. Box 3181
Lake Charles, La 70602-3181
Phone (318) 433-3690

**UniPlus** is a trademark of UniSoft Systems
**dBase III** is a registered trademark of Ashton-Tate
**Cromix-Plus** is a trademark Cromemco, Inc.
**dBIIICompiler** is a trademark of WordTech Systems, Inc.
**RealWorld** is a trademark of RealWorld Corp.

# dBIII Compiler

*Continued from front cover*

Our original article detailed many of the features that this data base compiler has. One should reference this first article if not familiar with the package. I will highlight the best features, the new features, and the improvements over some past weaknesses.

Since the file structures of both the .DBF and .NDX files are compatible with dBIII+, all files created on the Cromemco system can be transferred to an MS-DOS machine and run with the dBIII+ interpreter. We have tested files created on a Cromix system and downloaded to an AT&T 6300 running dBIII+ and have been able to use them with no problems.

The package uses the /etc/termcaps file for terminal capabilities. This enables the running of the same code on many different types of terminals. In addition, printed output can be piped through the spool utility.

One of the new features of Version 2 gives the programmer the ability to specify which spool commands the output is to use. In fact, the structure of the environmental variable that controls the piping can be used to pipe output into other Cromix utilities, such as type. This technique can be useful to dedicate an output device for program output.

Most of the new commands that are supported deal with multi-user and record locking schemes. The new commands are shown in **Figure 1.**

In addition, there are now four different locking schemes available at the time the compiled files are linked. The first two deal with locking of records upon reading them; the second two with locking of records on an update-only basis. Up to 10 work areas, each with up to 7 indexes, may be opened during program execution. Version 1 would require a second user just opening a .DBF file to sit and wait if the first user was still at the first USE command of the same file. This is no longer the case with Version 2. A file status variable that can detect locking problems is available to be used by the programmer.

Since this is a compiler, there are some dBIII+ commands which are not supported. These were detailed in our first article. However, all have programming work-arounds.

We presently have a number of applications running in this environment under Cromix. All of the data base manipulation is done on the Cromemco. Many reports are generated for analysis. However, we also download files to the MS-DOS machines when we have to use all of the interactive mode of dBIII+. It's a wonder why no one has come out with a package that would allow that MS-DOS machine a direct path to the Cromemco hard disk file to eliminate the downloading step.

If an error occurs at run-time, the program operator has an option to spool a status report to the system printer that details which files were in use at the time of the error, where the file pointers were at each file, what all the field variables contained in the current records, and the values of any memory variables. This is an extremely useful feature — especially in a multi-user system where much can be going on at one time.

An additional feature now found in Version 2 is a debugger. This program allows you to step through your application code one or more lines at a time. You can display memory or file values at any time, manipulate file operation, redirect printout to the system printer and trace variables. I have found this part of the package an extremely valuable time-saver in locating programming problems during the development stage.

Various minor bugs reported in Version 1 have been corrected. These included some indexing problems.

Version 1 came with a very thorough manual. The Version 2 manual is almost completed and will contain a special section on suggested multi-user programming techniques using this package.

For a cost of $995 this will give one an environment for true multi-user application programming, in a language well-supported in the MS-DOS world, at a reasonable price.

```
FLOCK()
RESTORE FROM {mfile} [EXCLUSIVE]
RLOCK()/LOCK()
SAVE
SET AUTOLOCK ON/OFF
SET DELAY TO {num}
SET EXCLUSIVE ON/OFF
SET INDEX TO [{filename}|{file list} [SHARED]]
SET MULTIUSER ON/OFF
SET RETRY TO {num}
SET SUSPEND ON/OFF
UNLOCK
USE [{filename}][INDEX {index name list} [SHARED]] [EXCLUSIVE]
     [ALIAS {alias name}]
USERNO()
```

Figure 1

# CS-460

*Continued from front cover*

To long-time Cromemco customers the most striking change in the CS-460 may be the new grey-tone color of the system. Cromemco has given the CS-460 a new, fresh look by departing from the brown-tone color scheme that has always been used in the 100 and 400-series equipment. The new "cloud grey" color is designed to match Cromemco's C-15 terminal and the new C-20 PC-compatible terminal.

All in all, the CS-460 is clearly another milestone in a long history of milestones from Cromemco. The CS-460 now replaces the CS-420 as Cromemco's flagship, and offers customers a new ultimate in system performance.

# NEW PRODUCTS...

*NEW PRODUCTS is a regularly appearing column devoted to announcing and following hardware and software products of interest to Cromemco users. Most information is derived from press releases submitted by vendors. As a result, I/O NEWS cannot be responsible for errors of ommission or any other inaccuracies.*

## CROMEMCO SYSTEM 460

Cromemco is an industry leader in the development of high performance computer systems. As with all our products the CS460 is designed for reliability as well as performance. Longtime customers, such as the U.S. Air Force, depend on Cromemco because our products consistently exceed their expectations of longevity and durability. Products are designed and can be manufactured to meet exacting military standards (such as MIL-STD 9858A, 480 and 483) further insuring product quality and integrity.

### Description

The System 460 is a high performance supermicrocomputer that offers an outstanding combination of speed, flexibility, rugged design and advanced packaging. It is designed to meet the rigorous demands of the multi-user, multi-tasking UNIX world. The CS460 is unique in its strong combination of minicomputer power and performance at supermicrocomputer prices.

The advanced design of the 32-bit CS460 extracts very high performance from both hardware and software. With its large system memory (up to 16 Mb), huge internal Winchester capacity, and ability to support up to 64 terminals/peripherals, the CS460 is designed to outperform traditional minicomputers. The CS460, combined with the industry standard UNIX V.2 operating system and a wide variety of hardware and software options, makes the CS460 the ideal computing engine in a variety of applications.

### Performance Plus

The CS460 utilizes the fast MC68020 operating at 16.7 MHz as its central processing unit (CPU). This 32-bit chip, combined with the MC68881 fast math chip operating at 25.0 MHz, enables the computer to perform over 1,300,000 Whetstones per second.

### Cache for Speed

The high operating speed of the System 460 is achieved largely through its cache-intensive design. Virtually every major system element — from CPU, to hard disk controller, to serial I/O controller — contains a local, high-speed buffer which serves to reduce bus traffic and increase operating speed. This extensive use of cache memory plays a major role in creating the CS460's high performance.

On the XXU processor board alone, Cromemco has placed a large 16 Kb of high-speed cache memory. This is 64 times that resident on the MC68020 chip and up to four times that found in many high performance supermicrocomputers available today.

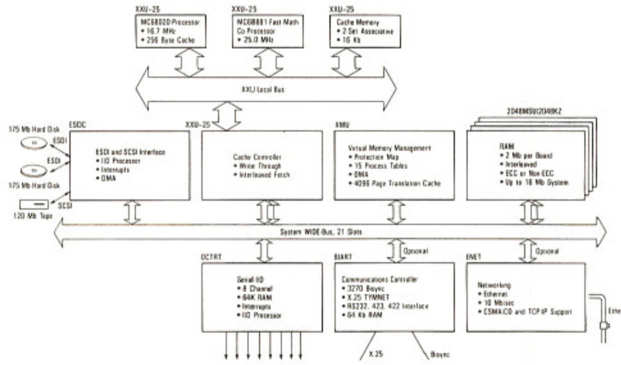
In addition to utilizing a large cache, Cromemco has made it two-set associative cache. Two-set associative cache is a state-of-the-art method of increasing cache efficiency. Less time is spent trying to input data to the cache memory resulting in faster task completion by the CPU.

This type of high performance cache is more commonly found in mainframe rather than supermicrocomputers. By implementing proven technology from larger systems Cromemco is providing a computing solution that is reliable as well as fast.

### Extensive High-Speed Memory

As with all other Cromemco systems, the CS460 supports a large quantity of system and peripheral memory. Memory can be expanded up to 16 Mb in 2 Mb increments. Both error correcting (ECC) and non-error correcting memory are offered.

SYSTEM DIAGRAM



As many as two hard disk drive units may be mounted internally, giving the CS460 up to 350 megabytes of internal storage. All disk I/O functions are supported by the ESDC hard disk controller board. The ESDC provides read-after-write verification, 196 Kb cache memory, and Direct Memory Access to the host address space. In addition, two ESDC boards in a system support overlapped seeks for increased data throughput in the dual drive configuration of the CS460.

### Comprehensive Diagnostics

Each time the CS460 is powered on, a comprehensive set of diagnostics, called XDOS, is executed. XDOS provides the operator with the operational status of the system. XDOS provides a thorough test of components including the CPU, memory, and I/O devices. It even checks itself by testing the ROM on which it is resident. Through XDOS and a modem it is also possible to perform remote diagnostics.

### Single-User, Multi-User

The standard CS460 is configured to support up to 8 terminals/peripherals through serial ports. This makes it ideal as a computing engine in software development, or as a distributed processing system. Additional OCTART boards, each supporting 8 channels, may be added to the system to provide up to 64 serial ports.

### The Power of UNIX

Whether your application requires FORTRAN programming for scientific purposes or a 4th Generation Language (4GL) for commercial applications, UNIX System V.2 provides the power and versatility that you need. Originally developed by Bell Laboratories, UNIX V.2 has wide acceptance as the standard operating system on high performance supermicrocomputers. Cromemco's UNIX V.2 adheres to industry standard enhancements, such as record and file level locking and TCP/IP. It supports such powerful features as Demand-Paged Virtual Memory, Common Object File Format (COFF), Job Control (Shell layering), and User cron Facility.

Included with Cromemco's UNIX are the highly acclaimed Berkeley enhancements, which include CSHELL, TERMCAP, and VI. Also included is ce, Cromemco's highly functional screen editor. This editor has all the power of VI, but is much easier to use.

### Internal 120 Megabyte Tape Back-up

The CS460 includes an integral 120 Megabyte Cartridge Tape Drive for fast data storage back-up and software interchange. This tape drive uses standard DC-600 A tape cartridges and records using the QIC-120 recording standard.

### User-Oriented Mechanical Design

The System 460 implements advanced design in the areas of cooling, power supply, and operator convenience. The front panel pilot lamp performs the triple functions of telling the operator that the system is on, that the air temperature in the box is at a safe level, and that all power supply voltages are within proper range. Similarly, an array of diagnostic indicators inside the box allow for an instant assessment of system status.

The airflow scheme in the System 460 is bi-level; separate thermal environments are maintained for the boards and peripherals. This allows the sensitive magnetic storage devices to operate without adverse effect from the heat produced by the logic boards. Twin thermal sensors feed information to a fan controller circuit which regulates the internal airflow.

A keylock on the power/reset switch ensures that a key operator can maintain control over the system. A second lock protects the card cage area and requires a separate key. This implements two levels of security over the data and physical components.

### Many Options

In addition to the CS460's high-performance, standard features, Cromemco offers many optional products that are designed to increase the system's functionality.

In the area of communications the optional BIART board supports both 3270 bisync for Micro-to-Mainframe communication, and X.25 the worldwide public data carrier network. Interface is accomplished through either RS-232, RS-422, or RS-423 serial channels.

Ethernet is supported with the optional E-Net board set and B-Net software package. This implementation supports CSMA/CD protocol in hardware and TCP/IP protocol in software.

Optional board products which support NTSC or PAL graphics, printers, peripheral, and other devices are also available.

### Extensive Software

Cromemco offers an extensive line of software to help the user take full advantage of UNIX's capabilities. For programmers, Cromemco offers a full line of development languages including C, FORTRAN-77, BASIC, and Pascal. An extremely capable 4th Generation Language, Informix 4GL is also available. This 4GL permits the development of applications in a fraction of the time required with conventional procedural languages. All software is supplied on convenient DC-600 tape cartridges.

**SYSTEM SPECIFICATIONS AND PRODUCT SUMMARY**

| MODEL | CS460EH175XX40 | CS460EH175XX40E | CS460EH350XX40 | CS460EH350XX40E |
|---|---|---|---|---|
| Processor | MC68020, 16.7 MHz | | | |
| Math Co-Processor | MC68881, 25.0 MHz | | | |
| Cache Memory<br>Processor (XXU)<br>Serial I/O (OCTRT)<br>Winchester Controller (ESDC) | <br>16 Kb<br>32 Kb<br>196 Kb | | | |
| RAM (expandable to 16 Mb) | 4 Mb | 4 Mb | 4 Mb | 4 Mb |
| Error-Correcting Memory (ECC) | No | Yes | No | Yes |
| ROM Firmware | XDOS diagnostics, with extensive system diagnostics | | | |
| Serial Interface | RS-232 or current loop, 8 channels, expandable to 64 | | | |
| Cartridge Tape Storage* | 120 Mb (Reads standard 60 Mb QIC-24 tapes or 120 Mb QIC-120 tapes) | | | |
| Hard Disk Storage* | 175 Mb | 175 Mb | 2-175 Mb | 2-175 Mb |
| Boards Supplied | XXU-25<br>XMU<br>OCTRT<br>ESDC<br>2-2048KZ | XXU-25<br>XMU<br>OCTRT<br>ESDC<br>2-2048MSU<br>MCUX | XXU-25<br>XMU<br>OCTRT<br>2-ESDC<br>2-2048KZ | XXU-25<br>XMU<br>OCTRT<br>2-ESDC<br>2-2048MSU<br>MCUX |
| Board Capacity | 21 Boards | | | |
| Open Board Slots | 15 | 14 | 14 | 13 |
| Operating Systems Installed | UNIX System V.2 | | | |
| Power | Operated from 100/115/130/220/240/260 volts, 50/60-cycle | | | |
| Power Consumption** | 500 watts | | | |
| Power Supply | +8 volts @ 30A, +16 volts @ 10A, −16 volts @ 5A | | | |
| Dimensions | 27½"H × 8½"W × 27½"D (69.9 cm × 21.6 cm × 69.9 cm) | | | |
| Weight | 115 lbs (52.3 kg) | | | |
| Mounting | Free-standing, floor; casters | | | |
| Operating Environment | 10–40°C | | | |

*Unformatted Capacity.

**Maximum power consumption based on all slots being used.

### WP WORD PROCESSING PROGRAM

AdriaSoft's WP™ is a visual-screen-to-printed-page word processor utilizing natural language commands. This "what you see is what you get" screen format facilitates editing, pagination and formatting; producing printed files just as they appear on the terminal screen. WP is loaded with features and designed to give you the ease of use you deserve.

### Terminal and Printer Flexibility

WP operates with a wide range of conventional terminals, including VT-100 and Cromemco's 3102, C-5, C-10, C-15, and C-20. An extended "termcap" terminal description database allows you to tailor the program to your specific needs.

In addition, WP offers flexible printer options, using laser, daisy-wheel and dot matrix printers. You can adapt the word processor to your own requirements using the "printcap" printer description database.

### UNIX and CROMIX Versions Available

WP is available for both Unix and Cromix (20 series and up). Path names are supported in Unix and Cromix allowing you to access files in other directories.

### Powerful and WriteMaster Compatible

AdriaSoft created WP for the 68000 CPU family to provide you a fast, powerful word processor which takes advantage of today's high performance computers. It is patterned after Cromemco's WriteMaster program so that the thousands of present WriteMaster users can move to WP without retraining. WP easily reads and writes files compatible with Cromemco's WriteMaster program.

*Continued on page 18*

# New Products

*Continued from page 17*

### Maximizes Your Hardware

AdriaSoft's WP utilizes the power provided by your 68000, 68010 or 68020 CPU. If your system is running with Cromemco's new XXU 68020/68881 processor board, WP takes advantage of the additional speed and cache memory.

### Easy To Use For Beginners and Experts

WP provides single keystroke help at all times and menu prompts for inexperienced users. Advanced users automatically bypass menus until needed.

### Selectable Page Formatting

The program provides a wide range of word processing features including: adaptable text alignment and justification; selectable end-of-line word wrapping; variable line spacing; automatic page numbering, page head and foot generation.

Tabs are completely flexible and user selectable. Document Alignment is also user selectable, by file, paragraph, cursor position or selected text. For tabular information or special emphasis, additional margins can be set within a file using Shift Margins and Lock so further alignment commands leave the inset copy intact. Unlock permits realignment of the inset copy.

Also included is a Merge command for personalized form letter generation. This feature provides easy addressing and placement of personalized information throughout the document. The Merge feature will also address mailing labels or envelopes from the identical merge database file used for the letters.

### Comprehensive Text Editing

WP offers many editing capabilities. Create Temporary files, Read one file into another or Write the current editing file to another file. You may Scan the current file or Type another file to view its contents while staying in the current editing file.

Find, searches for specified words or phrases or uses generic terms and categories. When Replace is used words and phrases can be changed repetitively, either semi or fully automatically. Jump and Go send the cursor to specific text points within a file. Index and Mark-for-Index allow index creation. When used within Find, an index can be added to a completed document.

Phrases or large portions of a document can be Moved, Copied, Deleted, Boldfaced or Underlined using the Select Text command. Superscripting and Subscripting characters, words or phrases is also easy with WP.

WP is available on 5-inch or 8-inch diskette for Unix or Cromix, priced at $995. To order, or for additional information please contact:

AdriaSoft
492 Glasgow Court
Milpitas, CA 95035
Telex (ITT) 493-3637
(408)262-2223

### THE BEST OF CDOS SOFTWARE — GRAPHICS

Applied Environmetrics is pleased to announce the release of **Volume 9** of the Best of Public Domain Software (PDS9) covering C-10 graphics for the terminal, for dot-matrix printers and for plotters.

The disk consists of three core programs: EASEL, PLOT33 and C10PLOT. EASEL allows simple access to the C-10 pixel graphics, and allows you to draw a picture on the C-10 screen using a 158 by 72 pixel resolution. The resulting picture, or large-lettering advertisement can be saved in a PIX file, and PIX files can be grouped together to make slide show (SHW) files. A demonstration program is provided.

PLOT33 is adapted from the well-known printer plotting programs on SIG/M 194. The programs have been translated into Cromemco Structured Basic, and can be used to produce plotting files. The disk is supplied with programs that plot directly onto an Epson, Citoh or Okidata printer but the source program PLOT33.Z80 is provided and can be assembled for plotting onto any other make of printer.

C10PLOT is a plotting package designed to produce line graphs, or scatter plots on a Graphtek MP1000 plotter.

All of the Best of Public Domain Software is guaranteed to work on a C-10 and has been tested under CDOS C2.65. Disks cost US$25 (A$37.50) each, and include air-mail delivery.

THE BEST OF PUBLIC DOMAIN SOFTWARE
Volume 1 — Games
(Colossal Cave Adventure, Sbasic Startrek, Trade, etc.)
Volume 2 — Communications
(Modem10, SQ—File squeezer and unsqueezer, etc.)
Volume 3 — Utilities
(Undelete, Catalogue, LU library utility, etc.)
Volume 4 — Assemblers
(8080 and Z80 assemblers and a loader)
Volume 5 — Disassembler
(Z80 Disassembler and a Tracer)
Volume 6 — Basic
(Tiny Basic, Sbasic Nolist cracker)
Volume 7 — Chess
(Fortran Source Code & Executable code)
Volume 8 — Genealogy
(dBase II programs)
Volume 9 — Graphics
(Plot 33, Easel, and C10Plot)

Applied Environmetrics
118 Gordon Street
BALWYN, 3103
Victoria, Australia

# Copyrights

*Continued from front cover*

3. Your name as an individual owner of the copyright interest (or your corporation name)

Without the above notice, the subject matter of your copyright may end up in the public domain, free for all to use in the U.S.

Most foreign countries do not require such a copyright notice. Nationals of these countries receive the benefit of their national copyright laws without any notice on their works. Many countries do not grant non-nationals the same rights as nationals.

### Universal Copyright Convention

However, if the work has a circle C notice, then the author is entitled to receive the same treatment as nationals in all of the countries that are members of the Universal Copyright Convention (UCC). The required circle C notice is the simplest version of the U.S. notice:

© 198? Your Name?

The word "Copyright" only has significance in English speaking countries, whereas the circle C symbol is recognized throughout the industrialized world under the Universal Copyright Convention (UCC). Merely by placing the circle C type notice on your software, you preserve your copyright interest around the world in all of the UCC member countries.

The table lists the 46 most significant countries and indicates their membership in various international copyright agreements. Very few countries are not members of the UCC. The U.S. has bilateral "side agreements" with most other nations.

The industrialized western countries historically have urged the developing third world countries to join the UCC and recognize copyrights (and patents and trademarks). This "urging" has intensified recently as international pirating of computer software (and hardware) increases. As a result, the list of UCC member countries is expected to increase each year. An updated UCC list (and other treaties) is available from the U.S. Copyright Office, Circular 38a (and b and c). Write to:

Copyright Office – Library of Congress
Washington, D.C. 20559

### Berne Convention ("Back Door")

The Berne Convention is very similar to the UCC and has almost the same nation membership. Only a few countries are members of Berne and not members of the UCC. The U.S. is currently not a member of the Berne convention. In order for U.S. nationals to obtain copyright protection in these Berne/non-UCC countries, they must use the "Back Door" approach. To extend your foreign rights to include the additional Berne /non-UCC countries you must publish in a Berne/UCC country (i.e., Canada or Mexico or the UK) simultaneously with (or within 30 days) of your U.S. distribution. Berne country publication permits U.S. nationals the benefit of the Berne Convention rights.

**Nation List — Treaties — Copyrights**

| | | | |
|---|---|---|---|
| Argentina | UCC g | Br | Bi |
| Australia | UCC gp + | Br | Bi |
| Austria | UCC gp + | Br | Bi |
| Bahamas | UCC gp | Br | |
| Benelux | UCC g | Br | Bi |
| Brazil | UCC gp | Br | Bi |
| Canada | UCC g + | Br | Bi |
| Chile | UCC g | Br | Bi |
| China PRC | | | Bi |
| China Taiwan | | | |
| Colombia | UCC gp | | |
| Costa Rica | UCC gp | Br | Bi |
| Czechoslovakia | UCC gp + | Br | Bi |
| Denmark | UCC gp + | Br | Bi |
| Ecuador | UCC g | | |
| Egypt | | Br | |
| Finland | UCC g + | Br | Bi |
| France | UCC gp | Br | Bi |
| Germany DemRep E | UCC gp | Br | |
| Germany FedRep W | UCC gp + | Br | Bi |
| Greece | UCC g + | Br | Bi |
| Hong Kong | | | |
| Hungary | UCC gp + | Br | Bi |
| Iceland | UCC g + | Br | Bi |
| India | UCC g + | Br | Bi |
| Ireland | UCC g + | Br | Bi |
| Israel | UCC g + | Br | Bi |
| Italy | UCC gp | Br | Bi |
| Japan | UCC gp + | Br | |
| Korea S | ?? | | |
| Mexico | UCC gp | Br | Bi |
| New Zealand | UCC g + | Br | Bi |
| Norway | UCC gp + | Br | Bi |
| Phillipines | UCC g | Br | Bi |
| Poland | UCC gp + | Br | Bi |
| Portugal | UCC gp | Br | Bi |
| Saudia Arabia | | | |
| South Africa. | | Br | Bi |
| Spain | UCC gp | Br | Bi |
| Switzerland | UCC gp + | Br | Bi |
| United Kingdom | UCC gp | Br | Bi |
| USA | UCC gp | | |
| USSR | UCC | | |
| Venezuela | UCC g | Br | |
| Yugoslavia | UCC gp + | Br | |

**UCC** = Universal Copyright Convention
**g** = Geneva 1952, **p** = Paris 1971
+ 1st Publication must be National
**Br** = Berne Con **Bi** = US Agreement

### Buenos Aires Convention

The phrase "All Rights Reserved" is commonly used on a second line of the notice. This phrase concerns the Buenos Aires Convention, and is required to obtain protection in Bolivia, Honduras, and Uruguay. The phrase is of little significance in most commercial situations.

# Tec Tip

### Ftar Caution

A standalone version of ftar is provided with Cromemco systems for the purpose of re-loading the operating system from either diskettes or from tape. After reading the last file with ftar it is critical that the operator depress carriage-return and wait 60 seconds for any information on RAM buffers to be written to the disk. This procedure is described in Appendix D of the Cromemco Introduction to UNIX manual.

# SOFT TOOLS

SOFT TOOLS is a regularly appearing column dedicated to UNIX and Cromix users. Its aim is towards simplifying the administration and maintenance of multi-user systems. It is edited by Tom Ronayne, President of Advanced Programming Techniques Corp. (APTC), P.O. Box 19549, Detroit, MI 48219, (313) 835-0808.

## Rounding

Rounding, and how to do it, is one of those slightly off-the-wall subjects that everybody tends to ignore until they get hit between the eyes with the problems that "trusting the compiler [interpreter]" can get you into.

When you round and how you round is important. The "when" is simple: you *never* round until the end of a calculation. Digital computing introduces enough pfudge-pfactor in calculations (there is, for example, no way to exactly represent 0.5 in binary) without adding to them by doing silly things — like intermediate rounding, so just make it a rule to not round anything but the last calculation's results and you'll (generally) be ok.

If rounding is done by compilers and interpreters, it's done in two "places:" when you either over- or underflow the word size in computations, and in the "formatted output" routines. Some compilers and interpreters "do you a favor" and round on over- and underflow. Whether this is really a favor is sometimes difficult to assess, and you can go on for hours arguing over which way is better and why.

Typically, if you're doing mixed-mode calculations, a floating point-to-integer conversion truncates (at least, you should hope that it truncates), so you don't really have to worry about mixed-mode.

Fun stuff can happen, though, when you're doing mixed-mode and single- and double-precision in calculations. You ought to be aware of what happens when you tell a compiler or interpreter to squeeze 64 bits into 32 (or 16, or 8), and vice-versa. Just remember, a rose by any other name may still be a rose, but single precision is different than double precision is different than integer.

If you look in your software manual, you'll see the range and precision of each number type, and bear in mind that when you, say, convert from double precision to single precision, you're either going to loose digits or they're going to be rounded up or rounded down, or truncated by the compiler in the conversion.

Sadly, most rounding that occurs in the world occurs by "round up on five;" i.e., if your answer is 5.245 and you want to round it to two decimal places, the result will be 5.25. My Texas Instruments calculators work this way, and, sorry to say, so do too many compilers and interpreters. The philosophy here is that you drop the first five digits (zero through four, and bump on the other five (five through nine). Good philosophy, bad practice.

There are a lot of folks in the world that can give you all the esoterics about why this is bad, but, I think, it breaks down to this: we do more calculations that wind up "half" than any others we do, and, we need to have a way to even out the "chance" that we'll round.

The American Society for Testing and Materials (ASTM) did so. ASTM E29-67 is the "Recommended Practice for Indicating Which Places of Figures are to be Considered Significant in Specified Limiting Values" (in English, that's how to round).

ASTM E 29-67 goes into a lot of detail about when, where, why, and how, which, if you're in the testing business, you should already know. We're going to limit the discussion to something important to most of us: money calculations. Note, though, that these routines are written so they can be used for multiple purposes, not just dollars and cents.

If you do money calculations — even something as simple as calculating sales taxes — you need to have a fair and predictable way of assuring that your calculations are rounded properly. You may have noticed something: the U.S. government doesn't make half and quarter pennies (I don't think other governments do either, but I stand to be corrected); point is, in North America, you can't pay a half penny, and you can't charge one, either.

Here (from ASTM E29-67) are the rules:

> "3.4.1 When the figure next beyond the last place to be retained is less than 5, retain unchanged the figure in the last place to be retained."
>
> "3.4.2 When the figure next beyond the last place to be retained is greater than 5, increase by 1 the figure in the last place to be retained."
>
> "3.4.3 When the figure next beyond the last place to be retained is 5, and there are no figures beyond this 5, or only zeros, increase by 1 the figure in the last place retained if it is odd, leave the figure unchanged if it is even. Increase by 1 the figure in the last place retained if there are figures beyond this five."

In English, if you've got 5.254 and you want to round it to two decimal places, it's 5.25; if you've got 5.256, it's 5.26; if you've got 5.255, you've got a problem to solve. The solution is: since the digit in the last place to be retained (the 5 in 5.25) is odd, you round it up to 5.26. What we're really doing here is a "fair coin" flip: you've got a fifty-fifty chance of having a head or tail (odd or even number), and you're making your rounding fair.

Is this important to you? Yes, if it's pennies you're talking about — pennies add up quickly when you do thousands of calculations (the reason why you don't ever round intermediate calculations). 'Thing is, you've got to be fair in both directions: in your favor, and in the other guy's. Also, if you do any work for government agencies (EPA, NHTSA, and others), this is the method you've got to use.

Now, you can find out real quick how your favorite language rounds. Just "formatted print" 5.255 to two decimal places; e.g., `write (*, '(f4.2)') 5.255` in Fortran, `printf ("%4.2f\n", 5.255);` in C, or `print using "#.##" 5.255` in Basic.

Try it: did you get 5.25 or did you get 5.26 (like you should)? Thought so.

**Listing 1** is a Rational Fortran (RatFor) function that, I think, is the "right" way, and **Listing 2** is a test program for it.

Here are the results from Listing 2:

```
Un-rounded Value 5.2500 Formatted 5.25 ASTM 5.2500
Un-rounded Value 5.2510 Formatted 5.25 ASTM 5.2500
Un-rounded Value 5.2520 Formatted 5.25 ASTM 5.2500
Un-rounded Value 5.2530 Formatted 5.25 ASTM 5.2500
Un-rounded Value 5.2540 Formatted 5.25 ASTM 5.2500
Un-rounded Value 5.2550 Formatted 5.25 ASTM 5.2600
Un-rounded Value 5.2560 Formatted 5.26 ASTM 5.2600
Un-rounded Value 5.2570 Formatted 5.26 ASTM 5.2600
Un-rounded Value 5.2580 Formatted 5.26 ASTM 5.2600
Un-rounded Value 5.2590 Formatted 5.26 ASTM 5.2600
```

''Sharpies'' will notice that Silicon Valley Software's (SVS) Fortran-77 compiler does it ''wrong:'' the formatted value of 5.255 is 5.25, not 5.26 like it should be. Those who actually test this, though, will notice (like I did) that SVS's C does do it right, the answer from C is 5.26.

If you need to use this ''astm'' routine with C, you're going to have to either figure out how to call a Fortran function from C, or you're going to have to invent your own tools for a power function and double-precision remaindering operations (which SVS C doesn't have).

No, we won't really do that to you.

**Listing 3** is the astm function in C. (Bye the by, I had trouble with this one, and if anybody's got a better idea, let me know.)

**Listing 4** is a power function for C, that calculates, say, y raised to the x power by the exp (ln (y) * x) method, where exp is the exponentiation function and ln is the natural logarithm function, both provided with SVS C.

Ha!, you say — there's a pwroften function provided with SVS C. Nope, sorry, won't work. In version 2.41 — which is what I have — the pwroften function returns zero for negative x's (which is either incredibly stupid or a big oversight on somebody's part); 10 raised to the -1 power is 0.10, not zero.

**Listing 5** is a remaindering function for C that calculates a double precision remainder by x1 - [x1 / x2] * x1, where [x] is the largest value less than or equal to x, (x2 != 0).

And, just for the heck of it, **Listing 6** is a single precision remaindering function for C. (Both of these are because C doesn't provide a modulus capability for real numbers, only for integers — the % operator.)

You should be able to find a use for the power, amod, and dmod functions whether or not you choose to use the astm routine.

If you'd like these routines, you can contact *I/O NEWS* for a disk, or to down-load from the IACU Cromix system, or you can get it from us.

You may dial-in to our Cromix-Plus system at 300, 1200, or 2400 baud at (313) 835-0809 and transfer the files to your system (we suggest you use the free ccall utility supplied with all Cromix systems). Or, you can send us a blank disk (5-1/4" or 8"),along with $5.00 (for postage and handling), and we'll copy everything for you. Be sure to tell us if you want CDOS, Cromix, UNIX, or MSDOS format, and number of sides and density (if you want to send a cartridge tape, that's fine, too — but be sure to tell us if it's an ''ftcd'' or ''ftp1'' format).

### LISTING 1: ASTM.RFR

```
#       astm--astm e29-67 rounding procedure
        real function astm (value, places)
        integer places                          # number of decimal places
        real value                              # value to be rounded
        double precision work, whole            # internals
#       magnitude of working value is 10 ^ number of decimal places
        work = 1.0d+1 ** places * value
#       whole number part of working value
        whole = dble (idint (work))
        if (work - whole >      4.999d-1)       # fake it
                whole = whole + 1.0d0
        else if (work - whole < 4.999d-1)       # fake it here, too
                ;
        else if (dmod (whole, 2.0d0) == 1.0d0)  # equal half: is it odd?
                whole = whole + 1.0d0
#       magnitude to original, convert to single precision
        astm = sngl (1.0d+1 ** (-places) * whole)
        return
        end
```

### LISTING 2: RASTM.RFR

```
#       tastm - test astm rounding function
        program tastm
        integer i, places
        real    value
        real    astm
        value  = 5.250                          # initial value for test
        places = 2                              # round to two places
        for (i = 1; i <= 10; i = i + 1) (
                write (*, "(a, f6.4, a, f4.2, a, f6.4)") "Unrounded Value ",
                        value, " Formatted ", value, " ASTM ", astm (value, places)
                value = value + 0.001
                )
        end
include astm.rfr
```

### LISTING 3: ASTM.C

```
/*      astm--astm e29-67 rounding procedure                                    */
double  astm (value, places)
double  value;
int     places;
(
        double  decimal, work, whole;
        double  power (), dmod ();

        decimal = (double) places;
/*      magnitude of working value is 10 ^ number of decimal places             */
        work = power (10.0, decimal) * value;
/*      whole number part of working value                                      */
        whole = (double) ((int) (work));
        if (work - whole > 0.5)                 /* greater than half            */
                whole += 1.0;
        else if (work - whole < 0.5)            /* less than half               */
                ;
        else if (dmod (whole, 2.0) == 1.0)      /* equal: is it odd?            */
                whole += 1.0;
/*      magnitude to original                                                   */
        return (power (10.0, -decimal) * whole);
)
```

### LISTING 4: POWER.C

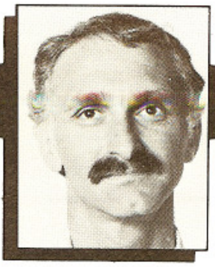
```
/*      power--calculate y^x using natural log and exponentiation       */
double  power (y, x)
double  y, x;
(
        double exp ();
        double ln ();

        return (exp (ln (y) * x));
)
```

### LISTING 5: DMOD.C

```
/*      dmod - analog of fortran double precision remaindering function */
double  dmod (xl, x2)
double  xl, x2;
(
        if (x2 == 0.0)                  /* avoid divide-by-zero error   */
                return (0.0);
        else
                return (xl - (long) (xl / x2) * x2);
)
```

### LISTING 6: AMOD.C

```
/*      amod - analog of fortran single precision remaindering function */
float   amod (xl, x2)
float   xl, x2;
(
        if (x2 == 0.0)                  /* avoid divide-by-zero error   */
                return (0.0);
        else
                return (xl - (int) (xl / x2) * x2);
)
```



# C-10 ENCOUNTERS

C-10 ENCOUNTERS is a regular column directed to users of Cromemco's personal computer, the C-10. It is edited by Dr. Tom Beer, of Applied Environmetrics, located at 118 Gordon St., Balwyn, Victoria 3103, Australia. Dr. Beer can be reached by phone during business hours at (03) 817-2571. Submit editorial directly to Dr. Beer.

## READERS QUERIES

Most of this column will be about spreadsheets. Before getting on to that I should mention that I have had a number of enquiries about the *Family History and Genealogy* software being sold by **Applied Environmetrics**. Readers want to know if they are going to have the same problems getting it up and running that were described in some of my previous columns. The good news is that the version of **Roots/M** being sold by Applied Environmetrics is the one that has been patched so as to work under CDOS. It definitely works fine on the C-10 and should also work well on any other machine using CDOS. It comes with a terminal installation program, so if you are not using the C-10 then the terminal settings can easily be altered to the terminal that you are using.

Mr. Gabriel Harris has written to me that he is arranging for the development of CDOS communications software that will allow Cromemco systems to talk to the Australian Video data base system called *Viatel*. This system operates on a split 75/1200 baud rate, whereas existing communications software suitable for Cromemco systems is geared to a single baud rate. If this is of interest to you then contact Mr. Harris at "Tregonning Hill", Stuart Town, N.S.W. 2820, Australia.

*I/O News Volume 6 No. 1* arrived today and in the C-10 encounters section there is written that "The C-10 will not support an underline cursor." Wrong. I must have been in a daze when I wrote that, and I am sure that many eagle-eyed readers are already penning letters of rebuke. An underline cursor is obtained by issuing the code Esc .Bi, with Esc .C for a blinking cursor, and Esc .D for a blinking underline cursor.

## SPREADSHEETS

An international telephone call from P. Honig in New York asking for advice about spreadsheets for the C-10 made me realise that this is a topic that has not yet been dealt with in these columns. I know of only two spreadsheets for the C-10. **Microsoft Multiplan** and the Cromemco spreadsheet that is provided on the C-10 systems disk.

The Cromemco spreadsheet started life as **Planmaster**, but with the Release 4 update of January 1984 it was renamed **Calcmaster**, and I will refer to it by this name even if I mean the earlier versions. It is made up of ten plansheets, each of which is limited to 31 rows and 13 columns. This means that if all the cells are full you can have 4,030 entries in Calcmaster. By contrast, Multiplan offers a single spreadsheet of 255 rows and 63 columns, giving 16,065 cells. The most annoying limitation of Calcmaster is that the user must design the plan so that he will never need more than 23 lines of calculation for each 13 x 31 page. One way of getting around this limitation is to pass the cell values to the next page and then define a further 23 equations. Thus technically one has the possibility for 230 calculations. It will, however, be very, very slow because Calcmaster seems to take an inordinate time to change pages.

Passing cell values across pages can be done by an equation of the form:

P2 = P1

which will transfer all the contents of page 1 to page 2, with the exception of the summations of line 31 and column 13. This seems obvious in retrospect but it took me quite a while to work out. Calcmaster users know that the manual is rather skimpy and that it is vital to read the online help file to work out what is going on. I had not, however realised that the help files had also changed with the various versions of the program. I should have guessed. The manual that I am using — the April 1983 edition — says on p.169 that functions are explained in the help file by typing d for define, then f for function. I was running Calcmaster 3.02, tried it, and it did not work. In fact the help file with Calcmaster 3.02 has less information than that of the earlier Planmaster help files, and it was only by going back to the earlier help files that I found out about P2 = P1.

The explanation of cell referencing also left me puzzled. The Calcmaster 3.02 help files tells us that in the reference L1(C3) the parentheses are necessary to indicate that it does not refer to a line and a column. This puzzled me as it seemed to indicate that Calcmaster actually understands something like L1C3. Well Calcmaster may understand it but I certainly do not. In testing it I found that when I was positioned on page 2, the definition C2C3 = C2(P1)C3(P1) seemed to act as if it were C1 = C2(P1) and L1C1 = L1(P1)C1(P1) seemed to act as L31 = L1(P1). I do not understand this behaviour and would welcome wisdom from any keen Calcmaster expert who can explain what is going on.

Those new to defining Calcmaster equations should also be warned that the equations must begin in column 1 and have no blank spaces anywhere in the line — and this means no trailing blanks. Appendix B of the manual tells us that Control-R removes characters but somewhat easier to remember is the fortunate fact that the Writemaster delete character (Control-9) will also get rid of the blanks. Not all Writemaster keystrokes work: delete endline does not.

Calcmaster has always had a fair share of problems. A January 1984 review in *Creative Computing* of the first release points out that it is "... a flaky program. It hangs, for example, if too many column widths are changed without saving the format." I was still finding similar problems with the fifth release (Calcmaster 3.02) which when operating under CDOS 3.07 would hang if I tabbed across to column 12. Cromemco themselves concede that the program has some peculiarities. Version 3.02 sends spurious characters when printing a plansheet that turn on underlining. The next time you print a plansheet, the spurious characters will turn off underlining. As I write this column there is a lawsuit pending against Lotus in which it is claimed that a user of Lotus 1-2-3 lost a large sum of money because there was a bug in Lotus 1-2-3 which produced incorrect results. Be warned. This can happen with Calcmaster. Cromemco concedes that in automatic calculation mode there is an option that rounds off calculations. When this option is selected, numbers may be rounded off incorrectly.

It is easy to get too carried away criticizing Calcmaster. But the truth is that it was supplied on the C10SP system disk at no extra cost and certainly, in my case, provided a piece of software that I would not have gone out and purchased. It was a great tool for learning about spreadsheets, and before I moved onto fancier spreadsheets it assisted in the production of two years of income tax data.

Multiplan uses a more sophisticated method of defining cell equations. Instead of having a list of equations like Calcmaster, Multiplan follows the Lotus 1-2-3 style of having an equation associated with each cell. I am sure that there must be a practical limit on the number and complexity of cell equations that Multiplan can handle and that this limit will be determined by the available C-10 memory. I do not happen to know what this limit is. Multiplan has many other nice features, for example it has a window function that allows you to view two far away columns on the same screen. All in all it would certainly be the choice for the serious C-10 spreadsheet user.

Since it first came out Multiplan has undergone a number of major revisions but my understanding is that only Multiplan Version 1 has been adapted to run on the C-10. Most dealers have stopped stocking old software like this but if you want it and are having trouble getting hold of it, or of TYPEQUICK for CDOS then I may be able to help. A problem is that you will be paying full price for software that is not the most up-to-date. Colin Marshall of the WA Cromemco Users mentioned a similar problem in relation to TYPEQUICK for Cromemcos. He contacted the producers of this program and was told that no version was available for CDOS. I find this surprising since a CDOS version was available as early as 1984 and I reviewed it in *I/O News Vol.4 #4*. All I can guess is that manufacturers have lost interest in small volume sales for operating systems outside of the mainstream.

Books on spreadsheets abound. I generally find it worthwhile to pop down to the local electronic shop that stocks TRS computers. In Australia these shops are called Tandy and my recollection is that TRS stands for Tandy Radio Shack. Because the TRS80 used a lot of similar software to the C-10 one can sometimes find relevant books quite cheap. Rodnay Zaks' book on programming the Z80, originally published by Sybex at an exorbitant price, was reprinted by Radio Shack as Cat. No. 62-2066 and sold for about one-third of the Sybex price. When I dropped in recently there was a copy of *The Power of Multiplan* (Radio Shack Cat. No. 62-1052.) for about $10. It seemed inexpensive enough so I purchased it, intending to work through its examples. Unfortunately I just could not muster up enough interest to do so. The book works on the presupposition that spreadsheet users are accountants and gave exercises in invoicing, accounts receivable, cost recovery, checkbook ledger, inventory, and so on. Very useful, but not very stimulating. I was far more interested in a book at another bookshop entitled *Mathematical Applications of Electronic Spread Sheets* by Deane E. Arganbright. (McGraw Hill, 1985) who shows how to use a spreadsheet to do algebraic and statistical calculations.

One of the most common statistical-type operations is linear regression. Here is a set of Calcmaster definitions that fit a straight line to data points in column 1 (x) and column 2 (y):

```
C3 = C1*C2
C4 = C1*C1
C5 = C2*C2
L31 = SUM(L1,L30)
C6 = 1
C6 = REPEAT(L2,L30,1 + C6(-1))
C7 = L31(C3)-(L31(C1)*L31(C2))/C6
C7 = C7/(L31(C4)-(L31(C1)**2)/C6
C8 = L31(C1)/C6-C7*L31(C2)/C6
```

You will be able to see that line 31 is being used for the sums of x, y, xy, x squared and y squared in the first five columns respectively. The problem to be overcome is that the linear regression formula requires the number of data points, N, and normally this would be entered in by the user. You would think that the computer could work out for itself how many data points there are, but even if you start out with a new spread sheet it is unable to tell whether the zeros are real data, or whether they represent no data. If both x and y equal to zero really represents missing data then this could be programmed into the define statement, but the absence of a count function in Calcmaster makes the task a bit difficult. The way I tackled it was to have column 6 give different possible values of N, and to calculate the regression coefficients in columns 7 and 8 such that the best straight line is:

y = C7*x + C8

For the values x = 26, 30, 44, 50, 62, 68, 74 and y = 92, 85, 78, 81, 54, 51, 40 the line of best fit, obtained from the numbers on line 7 of the spreadsheet, is:

y = -1.03x + 121.67

Once you get the hang of mathematical and statistical calculations on a spreadsheet it becomes quite addictive. CD

# INSIDE CROMIX

INSIDE CROMIX is an open forum on both eight-bit and 16-bit versions of Cromix. The subject matter is directed towards helping Cromix users derive more from their systems. Members' contributions are invited. INSIDE CROMIX is edited by Jordan Siedband, who can be reached at 5017 Fairview Lane, Skokie, IL 60077, (312)674-1175.

**Editor's Note:**

*It is with great sadness that I now tell you that this is the final installment of* Inside Cromix. *On January 11, 1988, Jordan Siedband finally succumbed to the cancer within him, and left us. Jordan's last gift to us, a distillation of the differences between 'C' programming for Cromix and Unix, is dedicated to his loving wife and son — may you know that our thoughts are with you. We will all miss him.*

Sooner or later, one must look at realities. Cromix is absolutely superb, but it is easy to see that CROMEMCO is drifting closer to the UNIX market and for good reason. There is much more support in the UNIX world. CROMEMCO is no longer the simple plug-in board manufacturer or the personal computer manufacturer but a high-tech great machine maker. It makes no sense to try to compete with the single-user market, even though us "old timers" feel abused.

I have had great fun/agony in converting a large collection of Z80C programs to Unix C. Everyone says that it is easy, but do not always believe them! However, a few suggestions may make it easier. The following are a few ideas which may help, and then again may not affect you at all.

1. If reading from a file fn, where n is integer, read(fn,&n,2); declare n short, Z80 int is 68000 short.
2. For getline() use gets().
3. Change exec() to execvp(). Be sure to declare:
   ```
   char *argo[] = {0,0};
   argo[0] = filename;
   execvp(filename,argo);
   ```
4. The time and date functions are all totally different.
5. In some line entry such as: printf("Enter 2.175 or 3-7/8"); where both are feasable. First try gets(buff), then parse, or scanf("%s",buff); one of these usually works.
6. To read a structure you MUST use the pointer: read(fn,&complex,16);
7. You must get used to the fact that fexec() does not exist, nor does all of that pretty record locking. UNIX is great because there is so much out there, but give me CROMIX for complete programmer control!!
8. Anything using system calls is almost certainly different from CROMIX. First look in the book indicated at the end of this article, then pray in a loud steady voice.
9. Math functions are generally not identical to their counterparts in Z80. Look in the Programming Guide.
10. Z80 CROMIX was great for financial calculations because it represented doubles as BCD. All of the 68000 C's whether CROMIX or UNIX use binary representation. If you need calculations to the penny, you must store x in cents (floor(100*x)). Do all of your arithmetic in cents but print in dollars = x/100.0. Floor is the equivalent of INT in BASIC.

Above all, don't panic!! Things that worked like a greased pig in Cromix may not work well in UNIX at all. It takes a steady hand and a lot of head bashing point where everything falls into place. A superb book to help you is:

*ADVANCED PROGRAMMER'S GUIDE TO UNIX SYSTEM V*
Rebecca Thomas, Lawrence Rogers, Jean Yates
McGraw-Hill 1986

(I suppose McGH will send me all kinds of books free for review — but this book is great for the programmer).

# 32K CLASSROOM

32K CLASSROOM is a regular column aimed at explaining programming techniques using Cromemco Structured BASIC. The column is edited by Bernie Thomas. Users are invited to submit programming examples and editorial contributions to Bernie Thomas at P.O. Box 50119, Nashville, Tennessee 37205.

I use a lot of screens which require the use of graphic characters and cursor location escape sequences. Some of these screens replicate the actual business form which is generated by the input, and some are quite complicated, and require quite a bit of code to create.

One such screen is a Receiving Form. When I first started using this screen, I included the code to create it in the programs in which it was to be used. This presented two problems. For one, it consumed lots of memory, and secondly, it was slow. Thanks to a friend, I learned to PRINT the output of the code to a file, and then GET the file as one giant string variable. To achieve this, the variable name which you use to GET the contents must be DIMensioned to exactly the size of the file. This technique prints the form on the screen many times faster than running the code. It still requires memory since the variable has to be dimensioned to the length of the file, which in the case of my receiving form is 1,664 bytes.

**LISTING 1** is the code which actually creates and builds the file which contains the Receiving Form. When developing a screen, I write the code as just @, and not @\1\, until it is complete and correct. When I am ready to actually create the file, I CHANGE the @'s to @\1\ using EDIT, and I leave out lines 10 through 50 until then.

This code works for Wyse and Televideo terminals. You must correct for your particular terminal.

After this program has been run, the following code will read the file and produce the screen.

```
100  Dim F$(1663)
110  Set 0,-1
120  Open\1\"recform.dat"
130  Get\1\F$
140  Close\1\
150  @ F$
```

This technique is infinitely better than running the code every time you wish to produce the screen, but there is even a better way yet. Since the recform.dat file is ASCII, you can use the TYpe command. This is a CROMIX Command, but it can be invoked from Basic using the methods I discussed in the last issue of *I/O NEWS*.

If you are using 68000 Basic, the following one-liner would produce the screen.

```
10  Sh "ty recform.dat"
```

If you are using the patched version of 32K as discussed in *Volume Two, Number Two* of *I/O NEWS*, the following would do produce the same results.

```
10    Dim A$(13)
20    A$ = "ty recform.dat"
30    X = Usr(%0125%,Adr(A$))
```

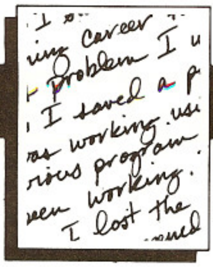
RAM Memory is not as big an issue today as it was when I began in 1975, but saving disk space certainly is. In the past, I have included all the routines which I use in a program such as the ones you see in lines 1000 through 1120 in every program which I wrote.

When you multiply the number of lines of code times the number of programs in the system it amounts to an enormous amount of disk space. This can be avoided by the use of Library Calls instead of Gosubs.

For those who are not using Library and Call, you may wish to refer to my article in *I/O NEWS Volume Five, Number Two*, and as I suggest in it, if you are not using these features, you should give them a try.

Listing 1

```
 10   On Error Goto Create'file
 20   Erase"recform.dat"
 30   On Error Stop
 40 *Create'file : Create"recform.dat"
 50   Open\1\"recform.dat"
 60   Set 0,-1 : Gosub Crt'clear : Gosub Screen'off
 70   Gosub Graph'on
 80   L=1 : C=1 : Gosub Loc : @\1\"2";
 90     For X=2 To 79
100     @\1\":";
110     Next X
120   @\1\"3";
130   L=2 : C=1 : Gosub Loc : @\1\"6"; : L=2 : C=80 : Gosub Loc : @\1\"6";
140   L=3 : C=1 : Gosub Loc : @\1\"6"; : Gosub Graph'off : Gosub Write'prot'on
150   C=5 : Gosub Loc : @\1\"RECEIVING NO "; : Gosub Undrlne : C=25
160   Gosub Loc : Gosub Limit
170   C=31 : Gosub Loc : @\1\"DATE "; : Gosub Undrlne : C=45
180   Gosub Loc : Gosub Limit
190   C=51 : Gosub Loc : @\1\"PURCHASE ORDER NO. "; : Gosub Undrlne : C=78
200   Gosub Loc : Gosub Limit
210   Gosub Graph'on : C=80 : Gosub Loc : @\1\"6";
220   L=4 : C=1 : Gosub Loc : @\1\"6"; : C=80 : Gosub Loc : @\1\"6";
230   L=5 : C=1 : Gosub Loc : @\1\"6"; : C=80 : Gosub Loc : @\1\"6";
240   L=6 : C=1 : Gosub Loc : @\1\"6"; : Gosub Graph'off
250   L=5 : C=3 : Gosub Loc : @\1\"FROM "; : Gosub Undrlne : C=39
260   Gosub Loc : Gosub Limit
270   L=6 : C=40 : Gosub Loc : @\1\"CARRIER "; : Gosub Undrlne : C=79
280   Gosub Loc : Gosub Limit
290   Gosub Graph'on : C=80 : Gosub Loc : @\1\"6";
300   L=7 : C=1 : Gosub Loc : @\1\"6"; : C=80 : Gosub Loc : @\1\"6";
310     For L=8.To 20 Step 2
320     C=1 : Gosub Loc : @\1\"4";
330       For X=1 To 78 : @\1\":"; : Next X
340     @\1\"9";
350     L=L+1 : C=1 : Gosub Loc : @\1\"6"; : C=80 : Gosub Loc
360     @\1\"6"; : L=L-1
370     Next L
380   L=22 : C=1 : Gosub Loc : @\1\"1";
390     For X=1 To 78 : @\1\":"; : Next X
400   @\1\"5";
410   L=8 : C=8 : Gosub Loc : @\1\"0"; : C=16 : Gosub Loc : @\1\"0";
420   C=24 : Gosub Loc : @\1\"0";
430     For L=9 To 21 Step 2
440     C=8 : Gosub Loc : @\1\"6"; : C=16 : Gosub Loc : @\1\"6";
450     C=24 : Gosub Loc : @\1\"6";
460     Next L
470     For L=10 To 20 Step 2
480     C=8 : Gosub Loc : @\1\"8"; : C=16 : Gosub Loc : @\1\"8";
490     C=24 : Gosub Loc : @\1\"8";
500     Next L
510   L=22 : C=8 : Gosub Loc : @\1\"="; : C=16 : Gosub Loc : @\1\"=";
520   C=24 : Gosub Loc : @\1\"=";
530   Gosub Graph'off
540   L=8 : C=3 : Gosub Loc : @\1\"QUAN"; : C=11 : Gosub Loc : @\1\"QUAN";
550   L=9 : C=3 : Gosub Loc : @\1\"REC'D"; : C=10 : Gosub Loc : @\1\"BCKORD";
560   C=17 : Gosub Loc : @\1\"SKU NO";
570   C=32 : Gosub Loc : @\1\"S I Z E  /  D E S C R I P T I O N"
580   Item=1
590     For L=11 To 21 Step 2
600     C=1 : Gosub Loc : @\1\Item; : Item=Item+1
610     Next L
620   Gosub Screen'on : Gosub Write'prot'off : Gosub Prot'on
630   @\1\ : Gosub Write'prot'off : Gosub Prot'off
640   Close\1\
650   End
660 *Crt'clear : @\1\Chr$(27);"*"; : Return
670 *Crt'scrub : @\1\Chr(27);":"; : Return
680 *Loc : @\1\Tab(0); : @\1\Chr$(27);"=";Chr$(L+31);Chr$(C+31); : Return
690 *Graph'on : @\1\Chr$(27);"H";Chr$(2); : Return
700 *Graph'off : @\1\Chr$(27);"H";Chr$(3); : Return
710 *Write'prot'on : @\1\Chr$(27);")"; : Return
720 *Write'prot'off : @\1\Chr$(27);"("; : Return
730 *Prot'on : @\1\Chr$(27);"&"; : Return
740 *Prot'off : @\1\Chr$(27);"'"; : Return
750 *Undrlne : @\1\Chr$(27);"G";"8"; : Return
760 *Limit : @\1\Chr$(27);"G";"0"; : Return
770 *Screen'off : @\1\Chr$(27);"`";"8"; : Return
780 *Screen'on : @\1\Chr$(27);"`";"9"; : Return
790 *Sav : Save"buildrecform.utl"
```



# USER NOTES

USER NOTES are useful techniques, tips, ideas and other helpful information gleaned from our member's experiences with their Cromemco systems. If you have something that you would like to share, write it up and send it to I/O NEWS, c/o USER NOTES, 24843 Del Prado, Suite 473, Dana Point, CA 92629-2852.

## Money From Heaven

**Dear Editor:**

The following program is something I wrote in SBASIC to have a game on my C-10 that small children could play. It's a simple "video-game" that is non-violent in nature: you catch money that is falling out of the sky. The speed of play is adjustable to accomodate differing skill levels. Kids seem to like it; some adults have spent a fair amount of time playing it also. To run it, simply type sbasic mfh.sav to CDOS or to the C-10 menu-prompt. I provide the source code for hackers who want to modify the game.

The normal repeat rate of the C-10 (and C-5) keyboard causes significant typeahead when the bucket-movement keys in the game are held down, resulting in an annoying "overshoot" problem. The keyboards repeat rate can be set manually to a more suitable value by pressing Control-Shift-D (see C-10 User Manual). As far as I can tell, there is no way to set the keyboard repeat rate from software; otherwise I would have done so.

Sincerely,

Jeff Johnson
1065 Lafayette #3
Denver, CO 80218
h: (303) 860-7705
w: (303) 930-5389

*[I tried the game and liked it! You do need to slow the keyboard repeat rate as explained above. As a welcome side-effect, the reduced keyboard repeat rate eliminated problems I was having with CE and WriteMaster, which had a tendency to display garbage characters if the cursor direction keys were held down when "motoring" through a text file. Ed.]*

```
100  REM  MONEY FROM HEAVEN -- Catch falling money with movable bucket

100  REM  Conventions used in this source file
100  REM    - SBasic reserved words in upper case (e.g., FOO)
100  REM    - Variables in mixed case, beginning lower (e.g., fooBar)
100  REM    - GOTO labels and subroutine and function names in mixed case,
100  REM      beginning upper (e.g., FooBar)
100  REM    - Blank lines separate meaningful segments of program
100  REM    - Subordinate statement blocks indented two spaces
100  REM    - Lines following unlabelled remarks have same line number
100  REM    - All source-editing done in Screen, not in SBasic
100  REM      (Note: SBasic List command changes formatting; do not use)

100  SET 0,-1
200  OPEN\1\"$CO"
300  RANDOMIZE
400  NOESC
500  DIM response$(100), chrs$(6), money$(0), beep$(0), underscore$(0)
600  INTEGER waitBetweenMoneyMoves, waitBetweenTrials, inputSlowdownFactor
700  INTEGER i, char, moneyX, moneyY, bucketX, score, scoreX, scoreY, missedMax
800  INTEGER nMissed, nMissedX, nMissedY, endMssgY, speed, speedX, speedY, byte
900  INTEGER leftLimit, rightLimit, speedUpperLimit, speedLowerLimit, moneyOnScreen

1000 REM  Need assembler routine into low memory to enable CDOS calls
1000 REM        pop bc  ;get cdos call indicator in c
1000 REM        call 5  ;call cdos
1000 REM        ld e,a  ;return value in de
1000 REM        ld d,0  ;/
1000 REM        ret     ;done
1000 DATA %00c1%,%00cd%,%0005%,%0000%,%005f%,%0016%,%0000%,%00c9%
1100 INTEGER CDOS, TestChar, GetChar: CDOS=%0103%: TestChar=11: GetChar=128
1200 FOR i=CDOS TO (CDOS+7)
1300   READ byte
1400   POKE i,byte
1500   NEXT i

1600 REM  Initialize "constants" (SBasic doesn't allow true constants)
1600 money$="$": beep$=Chr$(7): underscore$=Chr$(95)
1700 leftLimit=25: rightLimit=75: speedUpperLimit=9: speedLowerLimit=1
1800 endMssgY=10
1900 missedMax=5
2000 chrs$(1)=Chr$(8): chrs$(2)=Chr$(12): chrs$(3)=Chr$(27): chrs$(4)=Chr$(9)
2100 chrs$(5)=Chr$(11): chrs$(6)=Chr$(10)
2200 REM  Parameters to adjust speed (higher=slower), trials/game, etc.
2200 waitBetweenMoneyMoves=4: waitBetweenTrials=200: inputSlowdownFactor=4

2300 REM  Initialize variables
2300 speed=5
```

```
2400 *Start: PRINT\1,0\: PRINT\1,39\
2500 PRINT\1,1,1\Spc(20);"$$$$$$$$$$ MONEY FROM HEAVEN $$$$$$$$$$": PRINT
2600 PRINT "Catch falling money in bucket.": PRINT
2700 PRINT "LeftArrow key moves bucket left; RightArrow key moves bucket right.": PRINT
2800 PRINT "Speed is variable between 1 and 9."
2900 PRINT "UpArrow key increases speed, DownArrow key decreases it.": PRINT
3000 PRINT "The game ends when you've missed ";missedMax;" times.": PRINT
3100 INPUT "Press RETURN to begin.", response$

3200 REM  Set up display
3200 PRINT\1,0\

3300 REM                                                     scoreboard
3300 score=0
3400 scoreX=16: scoreY=1
3500 PRINT\1,1,scoreY\"Number Caught: ";score;
3600 REM                                                     miss counter
3600 nMissed=0
3700 nMissedX=16: nMissedY=2
3800 PRINT\1,1,nMissedY\"Number missed: ";nMissed;
3900 REM                                                     speed indicator
3900 speedX=8: speedY=24
4000 PRINT\1,1,speedY\"Speed: ";speed;
4100 REM                                                     bucket
4100 bucketX=leftLimit + (rightLimit - leftLimit)/2
4200 PRINT\1,(bucketX - 1),24\"\";underscore$;"/";
4300 REM                                                     walls
4300 PRINT\1,leftLimit - 2,24\"|";: PRINT\1,rightLimit + 2,24\"|";

4400 WHILE nMissed < missedMax
4500   REM  Wait before dropping money, but let user move bucket
4500   i=0: moneyOnScreen=NOT true
4600   REPEAT
4700     IF Usr(CDOS, TestChar) # 0 THEN DO
4800       GOSUB GetInput
4900       i=i + inputSlowDownFactor - 1
5000       ENDDO
5100     i=i + 1
5200     UNTIL i >= waitBetweenTrials
5300   REM  Drop money
5300   moneyX=Int((rightLimit - leftLimit) * Rnd(1) + leftLimit)
5400   FOR moneyY=1 TO 23
5500     PRINT\1,moneyX,moneyY\money$;
5600     REM  Timing Loop -- keep $$$ falling while handling user keystrokes
5600     i=0: moneyOnScreen=true
5700     REPEAT
5800       IF Usr(CDOS, TestChar) # 0 THEN DO
5900         GOSUB GetInput
6000         i=i + inputSlowdownFactor - 1
6100         ENDDO
6200       i=i + 1
6300       UNTIL i >= waitBetweenMoneyMoves * (speedUpperLimit + 1 - speed)
6400     PRINT\1,moneyX,moneyY\" ";
6500     NEXT moneyY
6600   REM  Did user catch or miss?
6600   IF (moneyX >= (bucketX - 1)) AND (moneyX <= (bucketX + 1)) THEN DO
6700     PRINT\1,(bucketX - 1),23\"\|/"
6800     score=score + 1
6900     PRINT\1,scoreX,scoreY\score;
7000     PRINT\1,(bucketX - 1),23\"   "
7100   ELSE
7200     PRINT beep$;
7300     nMissed=nMissed + 1
7400     PRINT\1,nMissedX,nMissedY\nMissed;
7500     ENDDO
7600   ENDWHILE

7700 PRINT\1,1,endMssgY\"Well, you missed ";missedMax;" times.": PRINT
7800 PRINT "Your score is shown above.": PRINT
7900 PRINT\1,38\
8000 REM  Gobble any leftover characters
8000 WHILE Usr(CDOS, TestChar) # 0
8100   char=Usr(CDOS. GetChar)

8200   ENDWHILE
8300 *Again: INPUT "Play again? (type y or n, then RETURN): ",response$
8400 IF response$(0,-1)="y" OR response$(0,-1)="Y" THEN GOTO Start
8500 IF response$(0,-1)="n" OR response$(0,-1)="N" THEN BYE
8600 PRINT "? Please answer y or n.": GOTO Again
8700 PRINT\1,38\
8800 BYE

8900 REM GetInput -- Subroutine to handle bucket movement and speed adjustment
8900 *GetInput: char=Usr(CDOS, GetChar)
9000   ON Pos(chrs$,Chr$(char),1) GOTO BucketLeft, BucketRight, BucketLeft, BucketRight, SpeedUp, SpeedDown
9100   GOTO GIdone
9200   *BucketLeft: REM User wants to move leftward
9300     IF bucketX > leftLimit THEN DO
9400       bucketX=bucketX - 1
9500       PRINT\1,(bucketX - 1),24\"\";underscore$;"/ ";
9600       ENDDO
9700     GOTO GIdone
9800   *BucketRight: REM User wants to move rightward
9900     IF bucketX < rightLimit THEN DO
10000      bucketX=bucketX + 1
10100      PRINT\1,(bucketX - 2),24\" \";underscore$;"/";
10200      ENDDO
10300    GOTO GIdone
10400  *SpeedUp: REM User wants to increase speed
10500    IF speed < speedUpperLimit THEN DO
10600      speed=speed + 1
10700      PRINT\1,speedX,speedY\speed;
10800      ENDDO
10900    GOTO GIdone
11000  *SpeedDown: REM User wants to decrease speed
11100    IF speed > speedLowerLimit THEN DO
11200      speed=speed - 1
11300      PRINT\1,speedX,speedY\speed;
11400      ENDDO
11500  *GIdone: RETURN
```

**Editor's Note:** *The following lp printer model was contributed by Carl Wick, 7082 P Road, Gladstone, MI 49837.*

**Editor:**

I have an IDS printer and wanted to use the printer functions completely, without having to add control characters to each file. Therefore I created the following print driver for lp. It may be possible for *I/O NEWS* to start a collection of these drives so that new system users could obtain a copy for their printers.

```
#       @(#)IDS          UniPlus V.2.1.2 (ATT 1.1)
# This dumb model has been modified to fit an prism ids 132  printer
# by C. J. Wick on 1987 March 14
# it set by the -o option on lp;  NOTE: -o maybe used more than once
# example -o-pl2 will give the user 12 point type
# type size are -pl0 = 10 point ; -pl2 = 12 point ; -pl6 = 16.8 point
# tabs are set by -on = tabs to be set ;
# paper width -w132 = 13.2 inches printed on ; w85 = 8.5 standard letter width ;
#               -w8.0 = 8.0 inches printer on
# -c = correspondence print mode ; -d = draft print mode
# Note: that the blank pages and "XXXXXXXXXX" on pages are comment out
# with '# *' to save paper and printer time
# lp interface for dumb line printer

# * x="XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX"
stty 9600 ixon -ixany -tabs 0<&1
# * echo "\014\c"
# * echo "$x\n$x\n$x\n$x\n"
banner "$2"
echo "\n"
user=`grep "^$2:" /etc/passwd | line | cut -d: -f5`
if [ -n "$user" ]
then
        echo "User: $user\n"
else
        echo "\n"
fi
echo "Request id: $1    Printer: `basename $0`\n"
date
echo "\n"
if [ -n "$3" ]
then
        banner $3
fi
copies=$4
# set default options
echo "\035\c"               # set to 10 point
echo "\033J,0,1584,$"       # set to 132 column paper
echo "\033B,6,$"            # set to 8 lines per in
echo "\033R,2,$"            # set to draft mode
for i in $5
do
        case "$i" in
        -c | c)         # correspondence print  (^[R,1,$)
                echo "\033R,1,$"
                ;;
        -d | d)         # draft print mode      (^[R,2,$)
                echo "\033R,2,$"
                ;;
        -pl0 | pl0)     # set character size to 10 point (^])
                echo "\035\c"
                ;;
        -pl2 | pl2)     # set charater size to 12 point (^^)
                echo "\036\c"
                ;;
        -pl6 | pl6)     # set character size to 16 point (^_)
                echo "\037\c"
                ;;
        -s6 | s6)       # 6 lines per inch (default mode)
                echo "\033B,8,$"                        # ^[B,8,$
                ;;
        -s8 | s8)       # 8 lines per inch
                echo "\033B,6,$"                        # ^[B,6,$
                ;;
        -w132 | w132)   # 13.2 inch width paper (^[J,1,1584,$)
                echo "\033J,1,1584,$"
                ;;
        -w85 | w85)     # 8.5 inch width paper  (^[J,1,1020,$)
                echo "\033J,1,1020,$"
                ;;
        -w80 | w80)     # 8.0 inch width paper  (^[J,1,960,$)
                echo "\033J,1,960"
                ;;
        -?)     echo "$i: not an option";
        esac
done
echo "\014\c"
shift; shift; shift; shift; shift
files="$*"
i=1
while [ $i -le $copies ]
do
        for file in $files
        do
                cat "$file" 2>&1
                echo "\014\c"
        done
        i=`expr $i + 1`
done
# * echo "$x\n$x\n$x\n$x\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n"
# * echo "\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n$x\n$x\n$x\n$x"
# * echo "$x\n$x\n$x\n$x\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n"
# * echo "\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n"
# * echo "$x\n$x\n$x\n$x\n$x\n$x\n$x\n$x\n"
sleep 120     # needed for printer buffer ( buffer size / characters per sec )
exit 0
```

**Editor's Note:** *The following patch was contributed by Alberic Muller, Rue du Jura 12, 2525 Le Landeron, Switzerland.*

There is a lot of interesting software in the public domain that is distributed on SSSD 8 inch floppy disks, for example from SIG/M, the CP/M User's Group, the C User's Group, etc. Many of these floppies were made on CP/M V2.2 which allows characters in the file name such as "@" or "-" that are illegal with CDOS. Cdoscopy will list these files but denies access to them.

An obvious solution is to go to a friend, boot his CP/M system and rename the file in a way consistent with CDOS. If you don't know anyone running CP/M or if you are in a hurry, what can be done?

The Cromix program patch will help. Start it and specifiy your floppy drive device:

```
$ patch /dev/fdc
```

Patch will prompt; now request to dump the device contents from absolute address 1900H. You will see that the file names are listed in the ASCII interpretation in the following way:

```
>d 1900
00001900: 00 2d 43 41  54 41 4c 4f - 47 30 31 38  00 00 00 0e  .-CATALOG018....
00001910: 02 03 00 00  00 00 00 00 - 00 00 00 00  00 00 00 00  ................
00001920: 00 40 44 20  20 20 20 20 - 20 43 20 20  00 00 00 20  .@D      C  ....
00001930: 04 05 06 07  00 00 00 00 - 00 00 00 00  00 00 00 00  ................
00001940: 00 40 44 20  20 20 20 20 - 20 43 4f 4d  00 00 00 5d  .@D      COM...]
00001950: 08 09 0a 0b  0c 0d 0e 0f - 10 11 12 13  00 00 00 00  ................
00001960: 00 43 48 41  4e 47 45 20 - 20 43 20 20  00 00 00 35  .CHANGE  C  ...5
00001970: 14 15 16 17  18 19 1a 00 - 00 00 00 00  00 00 00 00  ................
>
```

File name addresses, as can be seen, start at 1901H + (20H * N) where N = 0..63. There are 64 such directory entries. For example, to patch the second entry (N = 1), you would type the following command replacing '@' = 40H by 'A' = 41H:

```
>s 1921
00001921: 40 41
00001922: 44 .
>
```

We just changed the name '@D.C' to 'AD.C'. Type a dot to exit the loop. Not all the 64 entries can be patched that way; some of the larger files may require more than one entry to map their contents, so be sure to recognize their name, as listed in the Cdoscopy directory list.

Finally, exit patch with:

```
> ex
```

Exit with Control-C if you make a mistake, and patch will not rewrite the modified sectors. The addresses shown are valid only with Single-Side Single-Density floppies (26 sectors/77 cylinders).

# BITS & BYTES

BITS & BYTES is the place to look for the odd bit of information, opinion, programs, profiles and rumors that circulate through The IACU. Our ears are always attuned to any interesting miscellany — if you have something to contribute send it along to I/O NEWS • 24843 Del Prado, Suite 473 • Dana Point, CA 92629 •(714) 661-9764

## UNIX V.2 NOTES

*Source: Cromemco NEWS, Issue #126, Nov/Dec 1987.*

Cromemco's technical support group has compiled the following helpful notes on UNIX V.2 based on inquiries that they have received from customers:

**UNIX Mail:** In UNIX V.2, aliases require the mail program, mailx rather than the original mailer, mail. It is convenient in some cases to link rmail to mailx so that mail to and from remote systems using aliases are handled correctly.

**UNIX-U SUDS:** In the latest release of UNIX-U SUDS on disk, some directories were written with a path name that did not begin with "./". This has created a problem reading the disks with standalone ftar using the device designation "std(0,0)/" as stated in the manual. The device designation "std(0,0)./" will work in this case.

**TAKE/PUT Under UNIX:** Be sure to use the C shell (csh) when using put and take commands between two computers. The end of file is then ignored is then ignored when cu is disconnected to use put and take commands.

**UNIX V.2 Termcap/Terminfo:** To have terminfo and termcap work consistently for a particular terminal, be sure to use a terminal identifier that exists in both the termcap and terminfo libraries. For example, use the entry "c3102" for both termcap and terminfo.

**UNIX V.2 Using V.0 C Compiler:** An early release of the UNIX V.2 UPST included a copy of the cpp library for the V.0 compiler that was corrupted. The correct checksum of the library is 20107. Later versions contain the correct library.

**UNIX V.2, Release 2.01:** Normal default for the UNIX V.2 C compiler does not included floating point compilation and libraries. When a new environment variable FP is set (FP = 68881), the default changes to be floating point for the compiler. The loader, however, does not recognize the FP variable. Therefore, library tapes may be mixed if caution is taken. To assure proper operation of the C compiler, do not set the FP variable and use the standard documented instructions for compiling with floating point.

**UNIX V.2 Release 2.01:** The accounting programs may report the error message "/usr low on space" when no problem exists. The utility "devnm" used in the /etc/rc script used to recreate the mount table does not recognize new device path names such as /dev/std/0s0. Link /dev/std/0s0 to /dev/std0 to correct this problem.

If you have any questions about these or other aspects of your system operation, please feel free to contact the technical support group at Cromemco: (415)964-7400.

## Cromemco Announces Navy Contract

*Source: Cromemco News Release, 1/21/88.*

Cromemco, Inc. announced that the U.S. Navy has awarded a contract to a government systems integrator to supply Cromemco microcomputers, system software and support for the Supervisors of Shipbuilding, Conversions and Repair, USN (SUPSHIPS) Automated Information System (AIS) Modernization Program.

Various configurations of Cromemco's CS-420 microcomputers will be installed at 17 Navy sites throughout the world to provide SUPSHIPS with UNIX-based systems equipped with application software to manage the administration of construction, conversion, maintenance and repair of Naval ships.

## C-10 to Parallel Printer Fix

Jim Bailey, of Asgard Computing, contributed the following solution to a problem that was giving him trouble. If you use a C-10 and an 8-bit parallel printer, you may have encountered the same difficulty, namely, that when the printer is first turned on it will not respond when attempting to print a file. After turning the printer off, and then back on, everything works as it should. The cause of the problem has to do with the printer toggling the busy line high, and that the C-10 tests the busy line.

Jim's solution is a simple assembler program that brings the busy line low. You can either assemble this program using a Z80 assembler, or enter the corresponding byte values into an executable file using debug. Likewise, you could write an SBasic program to output the necessary bytes to an executable file (.com) or use poke and the usr function.

The program outputs a null character to port 50 (data port) and outputs 80H to port 68 (strobe). Two listings follow; the first is the assembly language version and the second is the byte values of the assembled program.

```
LD A,0     ;load null
OUT 50,A   ;output null to data port
LD A,80    ;load a 1 in high order bit
OUT 68,A   ;strobe line
JP,0       ;terminate
```

The assembled byte values are as follows:

```
3E 00 D3 50 3E 80 D3 68 C3 00
```

## Cromemco's WIDE-Bus ™

*Source: Cromemco NEWS, Issue #117, September 1986*

Cromemco customers have been impressed with the fantastic speed and performance of their System 420. They sometimes ask how Cromemco was able to achieve such speed with the S-100 bus. The answer is that they have extended the bus, keeping within the IEEE-696 standard, to a full 32-bits. They call it the WIDE-Bus™.

Cromemco's Word Interleaved Data Exchange Bus (WIDE-Bus) is a hosted bus that operates over the standard IEEE-696 bus. WIDE-Bus allows 32-bits of data to be transferred by utilizing interleaving techniques more commonly used on mainframe computers. The bus runs between the XXU processor board and the 2048KZ memory board. The MC68020 processor on the XXU generates queries to memory in 32-bit data packets. It then segments the query into two equal sized packets which it transmits over the WIDE-Bus in serial fashion. The 2048KZ accepts both packets of information, decodes them, and then locates the proper address in memory. Once the required data is located, the 2048KZ completes the cycle by transmitting it to the processor in dual 16-bit packets.

*Continued on page 30*

# BEST IN SALES AND SERVICE

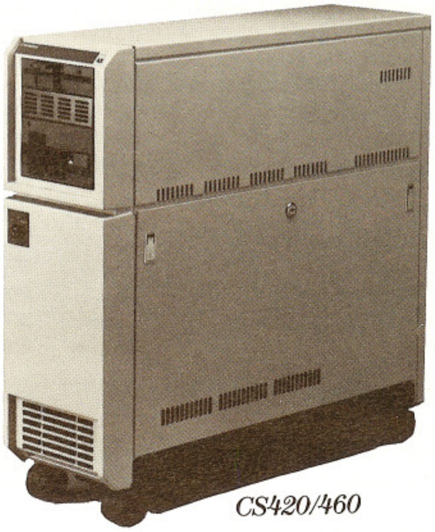
If you're looking for a local Cromemco dealer, search no more. **Computer Crossroads** is here, and ready to help you with the most qualified Cromemco sales and service team in the U.S.

Based in the greater Dallas area, we sell and service Cromemco equipment in all regions of the U.S. and in several foreign countries. Our central location and convenient access to DFW Airport means equipment can be received from any location in the U.S. and returned to the customer within three working days (excluding transit time), when our expedited service is requested.

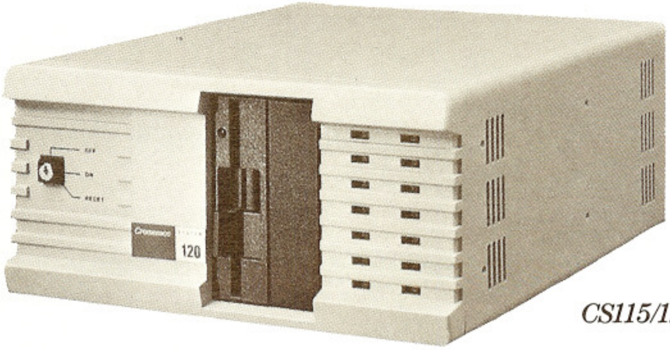
On top of that, you get the kind of expertise and support that has earned **Computer Crossroads** a host of Cromemco's "Outstanding Service" awards and the one and only "Number 1 U.S. Dealer" award. When it comes to quality, we can't be beat.

Service and support—more than just words—they are the reasons behind our phenomenal success. Simply said, sales backed by support means success and that means **Computer Crossroads**.

*MegaSafe™ Secure Data Cartridge*



*CS420/460*



*CS115/120/160*

**COMPUTER CROSSROADS of AMERICA, INC.**

SOFTWARE
SERVICE
HARDWARE

Where It All Comes Together

***Superb pricing on Cromemco equipment with the best in support at no extra cost.***

1750 Alma Road • Suite 118 • Richardson, Texas 75081

***FOR SUPER SERVICE CALL (214)231-6108 Telex: 4991118***

# Bits & Bytes

*Continued from page 28*

This technique allows Cromemco to maintain the growth path provided by the standard IEEE-696 bus, while at the same time matching the performance of systems utilizing such dedicated 32-bit buses as VME. The most widely noted implementation of this interleaving (or multiplexing) technique is on DEC's MicroVAX II. DEC's Q-Bus is a 16-bit bus which can transmit data in 32-bit segments using similar interleaving techniques as those implemented by Cromemco.

The implementation of WIDE-Bus combined with an integrated processor and fast floating point co-processor, and triple-cache architecture provides XXU systems such as the CS420 with the power to generate over 1,050,000 Whetstones (faster than the MicroVAX II) and over 3,700 Dhrystones (faster than the Sun 3/75). It is this performance that continues to attract results-oriented users to Cromemco computers.

**Technical Support News**

*Source: Cromemco NEWS, Issue #125, Sep/Oct 1987*

**MAXPMEM:** The MAXPMEM variable may be used to solve a subtle problem encountered under the UNIX V.2 Operating System. When the amount of system memory is greater than 8 megabytes, the Operating System may incorrectly determine the amount of system memory available.

If the MAXPMEM variable is set in the kernel to 0, the UNIX V.2 Operating System will automatically configure the amount of system memory available by checking the installed memory.

Case One: Graphics memory normally starts at an 8 megabyte address location. If the system memory is 8 megabytes, then the automatic memory configuration assumes incorrectly that the graphics memory (starting at 8 megabytes) is part of the system memory. Therefore, when a system is heavily loaded, programs may be run in graphics memory.

The amount of system memory should be defined by setting the MAXPMEM variable in the kernel. This may be changed in the /etc/master file and the kernel remade. MAXPMEM is the number of 4K pages of memory. If more than 8 megabytes of system memory is installed, the starting address of graphics memory should be moved above the last page of system memory. The highest starting memory address that can be used for the last page of graphics memory is E00000 hexadecimal.

Case Two: The top 1.5 megabytes of the addressable 16 megabytes of system memory is mapped for I/O and common data usage by the operating system. When physical system memory is greater than 14 megabytes, MAXPMEM must be set to limit the physical memory usage below 14.5 megabytes. (MAXPMEM is less than or equal to E7F hex or 3711.)

Note that UNIX V.0 does not autoconfigure the system memory. The amount of available system memory is set with the MAXMEM1 variable. The default is 2 megabytes of memory and is defined as the number of 4K pages of memory. MAXMEM1 can be changed on UNIX V.0 systems with configuration software or by using the ADB utility.

**MAXUMEM**: Another V.2 Operating System variable that is helpful to know is MAXUMEM. This sets the maximum total memory available for all processes run by any single user. This must be large enough to accommodate not only the program memory usage but it must also account for the memory accessed by the PHYS system call. The error that occurs from this limitation is "not enough memory to load."

# Output

*Continued from page 5*

also work harder to get the magazine out on a bi-monthly schedule.

As individuals you can do even more. For starters, if your membership is up for renewal, renew it! Enlist friends and colleagues into the association. If your company has one subscription to *I/O NEWS*, but there are more than three people sharing it, talk to us about getting a company membership at a discount rate.

And while you're at it, why not contact some of the companies that advertise their products and services in the magazine and let them know, if nothing else, that you're glad that they are still involved with Cromemco, and that you saw their ad in *I/O NEWS*.

Let's pull together. We need to do it now, more than ever.

Bill

Bill Jaenicke
Editor

FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST• FAST

# PERFORMANCE PLUS

## See Wordstar™, dBASEII™, and other Z80 software really fly!

## SLAVE PROCESSORS FOR 11.27 or 31.05 CROMIX

8MHZ Z80H 256K

TWO USERS PER SLAVE

Increase Performance Two to Fourteen Times

Much Faster Terminal I/O

$1095.00 includes software license

(Also ideal for fast process control applications)

## NEW 32K STRUCTURED BASIC

Formatted input — High speed built-in Sort

Programs load up to FIVE TIMES FASTER

| | |
|---|---|
| CROMIX (SBASIC.BIN) | $295 |
| CP/M | $195 |
| UPDATE CROMIX VERSION | $95* |
| UPDATE CDOS VERSION | $95* |

* Must be holder of valid Cromemco License

## RUN YOUR FAVORITE SBASIC PROGRAMS ON PC/MS-DOS SYSTEMS

| | |
|---|---|
| SBASIC.PC | $295 |
| SBASIC.PC RUNTIME | $100 |

**SYSTEMS ATLANTA, INC., P.O. BOX 99, LEBANON, GA 30146** **(404) 928-0240**

*Wordstar is a trademark of Micropro. dBASEII is a trademark of Ashton-Tate.* **Fax: (404) 928-9396**

•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST• FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•FAST•

# Now! Upgrade Your Cromix or Unix System to

# 1,050,000

# WHETSTONES PER SECOND

## FASTEST CPU RATE YET

Now you can soar to the outer limits of modern software capability at processing rates never before possible. Upgrade your current Cromemco to the speed of our newest systems with the remarkable XXU processor board. It combines the proven Motorola 68020 chip and the Motorola 68881 co-processor into a single unit with integral high speed cache memory -- larger and faster than any other computer in its class. These figures tell the story.*

| MANUFACTURER | MODEL | WHETSTONES/SEC. |
|---|---|---|
| **CROMEMCO** | **XXU Based** | **1,050,000** |
| DEC | MICROVAX II | 887,000 |
| SUN | 3/50 | 860,000 |
| APOLLO | 3000 | 780,000 |
| DEC | VAX 11/780 | 476,000 |
| IBM | PC/RT | 200,000 |

*Test data obtained from Datamation and Unix World magazines.

## LOW COST

As little as $4995 will convert your recent model Cromemco system to this high speed capability. Older models will cost somewhat more, but still less than half the cost of a new system. Low purchase price, however, is only the tip of the iceberg. The big cost savings come into play once you've started using your XXU-based system.

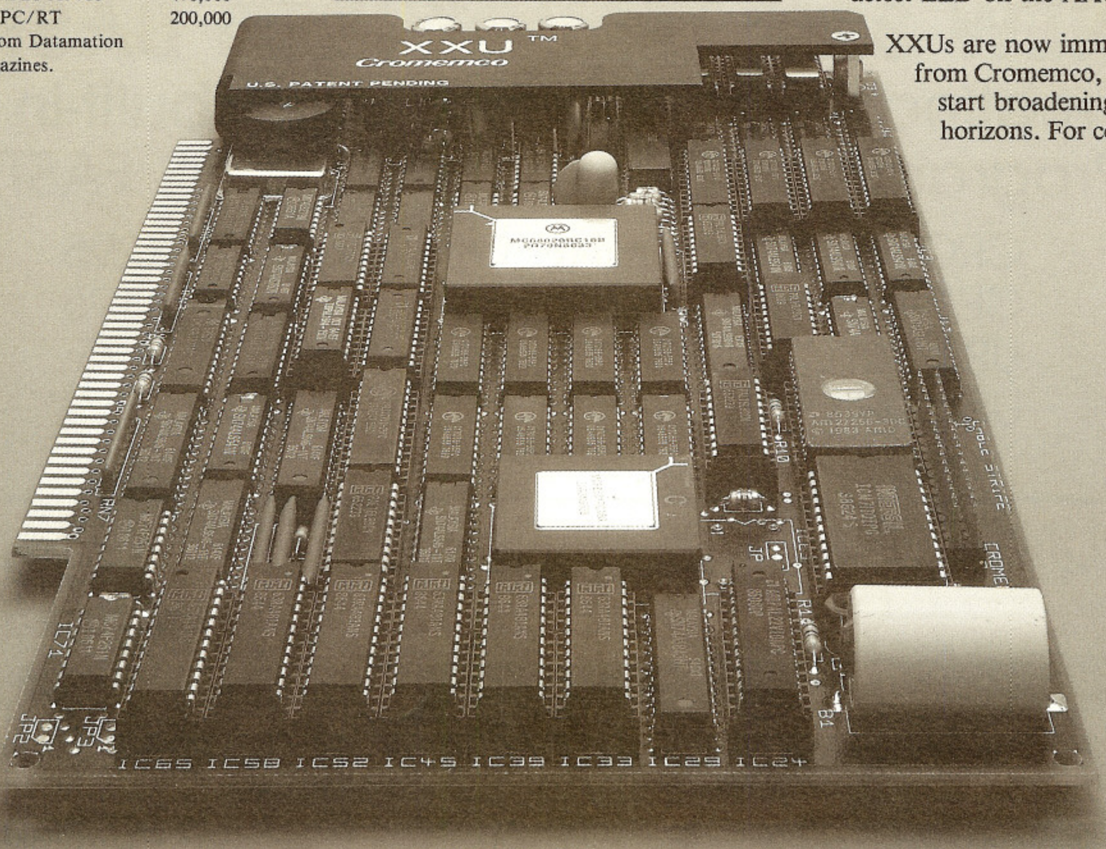
# XXU

## EASY UPGRADING

Depending on your model of Cromix or Unix system, upgrading may be as simple as plugging in a single board. Older systems can be easily upgraded by your dealer or at the factory. Either way, your new XXU is fully backed by Cromix and Unix software support and our factory warranty.

## PLUS THESE ADDED CONVENIENCE FEATURES

A real time clock/calendar, powered by a seven year lithium cell battery, eliminates time and date setting tasks. The on-board XDOS/boot ROM program performs system diagnostics and controls a fault detect LED on the XXU board.



XXUs are now immediately available from Cromemco, so order now and start broadening your processing horizons. For complete data, call or write today.

Cromemco
A DYNATECH COMPANY

CROMEMCO, INC.
280 Bernardo Ave., P.O. Box 7400
Mountain View, CA 94039
(415) 964-7400

**IN EUROPE:** Cromemco GmbH
6236 Eschborn 1, Frankfurter Str. 33-35
P.O. Box 5267, Germany
0049 (06196) 481606